

GELE3700
Projet de génie électrique 1

Rapport Final

Par :

Andrée-Maude DeGrâce – A00167485
Samuel DeGrâce – A00173226
François-Guillaume Landry – A00160020
Khadiétou Mounirou Ndiaye – A00176226
Aboubacar Rabiou Ousman – A00176220

Présenté à :

Professeur Gabriel Cormier
Ph.D., Ing.

2017-04-18



**FACULTÉ
D'INGÉNIERIE**

**UNIVERSITÉ
DE
MONCTON**

Moncton, NB, Canada

Sommaire

L'objectif de ce projet d'ingénierie est de faire la conception et la création d'un système d'acquisition de données. Le système sera utilisé lors d'une recherche menée par Dr. Grant Handrigan de l'école de kinésiologie et de loisirs de l'Université de Moncton. L'étude vise à comparer l'effet de l'entraînement avec et sans souliers sur l'endurance musculaire. Les données seront amassées à l'aide du capteur MPU 6050 qui contient un accéléromètre et un gyroscope. Ces données seront recueillies à l'aide d'une unité à microcontrôleur pour ensuite être envoyées via Wi-Fi à un code Java où elles seront traitées puis emmagasinées dans une base de données MySQL. Le traitement des données se termine avec un code Matlab qui est en charge de l'affichage des données à l'aide de graphiques et de tableaux. Le système doit être portatif et assez petit pour être attaché sur un athlète qui effectuera une course de 30 minutes. Des connaissances en génie électrique, logiciel et mécanique seront nécessaires pour mener ce projet à terme.

Remerciement

Nous tenons à remercier dans un premier temps Monsieur Gabriel Cormier, Ph.D., Ing., SMIEEE, sans qui la réalisation de ce projet ne se serait pas autant bien déroulée. Toute l'équipe est très reconnaissante de l'aide offert par M. Cormier et de son enthousiasme par rapport à notre projet.

Nous souhaitons également remercier Monsieur Jonathan St-Pierre, technicien de laboratoire, qui nous a assisté pour l'impression de quelques composantes de notre projet ainsi que Monsieur Yassine Bouslimani Ph.D., Ing., qui nous a guidé lors de la programmation Wi-Fi. Nous voulons de même remercier Monsieur Grant Handrigan, PhD, pour nous avoir offert l'opportunité de réaliser ce projet.

Finalement, nous voulons remercier l'Université de Moncton, campus de Moncton, ainsi que la faculté d'ingénierie du campus pour le financement de l'achat de nos pièces ainsi que pour le matériel fourni dans les laboratoires.

Table des matières

Sommaire	i
Remerciement.....	ii
Table des matières.....	iii
Liste des figures	vi
Liste des tableaux.....	viii
Chapitre 1 : Introduction	1
1.1 Description de l'étude pour laquelle le produit sera conçu	1
1.2 Cahier de charges du client	2
Chapitre 2 : Recherches et études préliminaires.....	3
2.1 Étude de faisabilité.....	3
2.1.1 Résumé des besoins	3
2.1.2 Défauts des appareils actuellement disponibles	3
2.1.3 Objectifs de travail	6
2.1.4 Cahier de charges	7
2.1.4.1 La conception physique	7
2.1.4.2 Le recueil des données	7
2.1.4.3 L'alimentation de l'appareil.....	8
2.1.4.4 L'aspect environnemental	8
2.1.4.5 Le coût de fabrication.....	8
2.2 Diagramme fonctionnel.....	10
2.3 Présentation des concepts	11
2.3.1 Microcontrôleur	12
2.3.2 Affichage	13
2.3.3 Batterie	13
2.3.4 Communication	14
Micro-SD	14
Sans fil (Bluetooth et Wi-Fi)	14
2.3.5 Capteurs.....	14
2.4 Étude de faisabilité.....	16
2.4 Matrice de décision	17
2.4.1 Explication	19

2.5 Échéancier	20
Chapitre 3 : Design de l'appareil	21
3.1. Circuit intégré	21
3.2 Boîtier	23
3.3 Développement du logiciel de système embarqué	24
3.3.1 Mesures d'accélération et gyroscope	26
3.3.2 Envoi UDP	27
3.3.3 Alimentation et LED	28
Chapitre 4 : Développement du logiciel pour le recueil et l'analyse des données	31
4.1 Architecture logicielle du projet	32
4.2 Gestion du développement avec GitHub	33
4.3 Choix de la base de données	35
4.4 Présentation des codes pour essais préliminaires	35
4.4.1 Recueil des données du microcontrôleur sur le port série	35
4.4.2 Réception UDP sur Matlab	39
4.5 Présentation du concept final	40
4.5.1 Création et utilisation d'une base de données sur MySQL	40
4.5.1.1 Installation	40
4.5.1.2 Connexion et déconnexion du client	41
4.5.1.3 Création d'une base de données	41
4.5.1.4 Création d'une table	42
4.5.1.5 Ajout de données et consultation de la table	43
4.5.2 Code Processing pour le recueil de données et l'envoi vers MySQL	44
4.5.2.1 Réception par protocole UDP	46
4.5.2.2 Envoi d'une requête vers MySQL	48
4.5.3 Design de la GUI sur Matlab	48
4.5.4 Présentation du code Matlab	51
4.5.4.1 Fonction requeteSQL()	52
Chapitre 5 : Résultats	53
5.1 Boîtier et circuit imprimé	53
5.2 Logiciel sur système embarqué et envoi UDP	53
5.3 Réception UDP avec Processing et envoi vers la base de données	54
5.4 Interface graphique	55

5.5 Alimentation et batterie.....	57
Chapitre 6 : Conclusion	59
6.1 Recommandations	59
6.2 Conclusion	59
Reference	61
Chapitre 7 : Annexe.....	62
7.1 Spécifications de la batterie	62
7.2 Codes sur GitHub.....	62
7.2.1 Code Arduino	62
7.2.2 Code Processing.....	62
7.2.3 Code Matlab	62

Liste des figures

Figure 1 : Arbre des objectifs.	6
Figure 2 : Diagramme fonctionnel	10
Figure 3- Connexions sur le circuit imprimé	21
Figure 4 : Connexion sur le circuit imprimé	21
Figure 6 : Différentes vues du boîtier	23
Figure 7 : Conception finale 3D du boîtier	24
Figure 8 : Ordinogramme du code de système embarqué.....	25
Figure 9: Montage du circuit de la batterie et de la LED.	29
Figure 10 : Couleur de la LED en fonction du niveau de tension.....	29
Figure 11 Organigramme du programme d’affichage d’état de la batterie.....	30
Figure 12 : Diagramme fonctionnel avec encadré des fonctions reliées au développement logiciel.....	31
Figure 13 : Schéma bloc des composantes logicielles	32
Figure 14 : Aperçu de GitHub lorsque le projet était encore en phase de développement (2 avril 2017).....	34
Figure 15 : Envoi de données de température (capteur LM35) sur un port série et enregistrement dans la base de données.....	36
Figure 16 : Code Arduino qui recueille les données de température du capteur LM35	37
Figure 17 : Script Python pour le recueil de données sur le port série	37
Figure 18 : Interface graphique provisoire sur Matlab. L’image du haut montre les données de température recueillies à un temps t_0 . L’image du bas montre les données de température recueillies à un temps t_1 , après avoir appuyé sur le bouton “requête MySQL” et ensuite sur le bouton “visualiser”	38
Figure 19 : Client UDP implémenté sur Matlab	39
Figure 20 : Erreur de délai lors de la réception UDP sur Matlab	40
Figure 21 : Format de la table utilisée pour stocker les données.....	43
Figure 22 : Ajout d’une donnée dans la table et réponse.....	44
Figure 23 : Affichage de la table	44
Figure 24 : Ordinogramme du code Processing pour la réception UDP et l’envoi vers MySQL	45
Figure 25 : Exemple d’affichage des données sur la GUI Processing	47
Figure 26 : Insertion d’une donnée mesurée dans la base de données MySQL.....	48
Figure 27 : Conception de l’interface graphique sur GUIDE.....	49
Figure 28 : Ordinogramme du code de la GUI Matlab.....	51
Figure 29 : Prototype de l’appareil utilisé pour tester les codes.....	53
Figure 30 : Données envoyées par le système embarqué sur le port série	54
Figure 31 : Exemple de fonctionnement de la réception UDP et de l’envoi vers MySQL	54
Figure 32 : Test de la GUI avec une prominence minimale des pics de 40	55

Figure 33 : Diminution du nombre de pics avec une augmentation de la prominence minimale	56
Figure 34 : Identification des pas en utilisant une prominence minimale de 200	57

Liste des tableaux

Tableau 1 : Réponses aux questions de base pour l'identification des besoins.....	3
Tableau 2 : Défauts de produits actuellement sur le marché	5
Tableau 3 : Cahier de charges	9
Tableau 4 : Concepts proposés	11
Tableau 5 : Choix de la batterie	13
Tableau 6 : Caractéristiques du MPU6050	15
Tableau 7 : Caractéristiques du LSM6DS3	15
Tableau 8 : Caractéristiques du 9DoF Sensor Stick.....	16
Tableau 9 : Analyse des concepts	17
Tableau 10 : Matrice de décision.....	18
Tableau 11 : Échéancier initial du projet	20
Tableau 12 : Calibres du capteur MPU-6050 (gyroscope et accéléromètre)	26
Tableau 13 : Trame d'envoi des données	27
Tableau 14 : Encodage de la trame envoyée sur UDP	47
Tableau 15 : Spécification de la batterie (PKCELL LP503562).	62

Chapitre 1 : Introduction

Le produit à réaliser dans le cadre de ce projet devra permettre de recueillir des données cinématiques associées aux foulées lors de la course à pied. L'appareil devra servir en tant qu'instrument de mesure dans une étude scientifique réalisée par le Dr. Grant Handrigan de la Faculté de kinésiologie de l'Université de Moncton, campus de Moncton. Entre autres, l'appareil servira à recueillir et afficher les données brutes d'un capteur d'accélération installé sur un athlète pratiquant la course. L'étude effectuée par le laboratoire de G. Handrigan porte sur les effets de courir avec et sans soulier. Un appareil pouvant capter des valeurs d'accélération linéaire et angulaire et les afficher sera donc l'objectif à réaliser pour ce projet. L'appareil devra bien sûr être le plus confortable et léger possible pour maximiser les performances de l'athlète en plus d'être le moins dispendieux possible.

1.1 Description de l'étude pour laquelle le produit sera conçu

Le but principal de l'étude est d'évaluer l'effet d'un entraînement avec pieds nus versus un entraînement avec souliers chez des athlètes de haut niveau qui pratiquent le cross-country. Les participants à l'étude seront 10 hommes et 10 femmes de l'équipe de cross-country de l'Université de Moncton. L'hypothèse avancée par les chercheurs est que l'entraînement avec pieds nus augmente la performance et permet ensuite aux athlètes de courir la même distance à même vitesse en utilisant moins d'énergie. Les objectifs de l'étude en lien avec la mesure de données cinématiques sont les suivants :

- Est-ce que la fréquence des foulées augmente en courant pieds nus ?
- Est-ce qu'il y a une différence d'efficacité entre la course avec souliers et la course avec pieds nus ?
- Quels sont les patrons de mouvements susceptibles de diminuer les taux de blessure ?

L'étude comportera trois séances où chaque athlète devra effectuer une session d'entraînement typique de coureur d'endurance. Au début, un test d'efficacité de course sera réalisé : il consiste en trois essais aléatoires de huit minutes à différentes vitesses. Par la suite, les participants effectueront un test de fatigue au stade intérieur du CEPS de l'Université de Moncton consistant en sept séries de 1000m (à 94-97% de leur vitesse aérobie maximale). Un test d'efficacité de la course sera effectué une dernière fois à la fin de la session. Au total, chaque séance durera environ une heure et demie.

1.2 Cahier de charges du client

Le dispositif à concevoir devra être porté par les athlètes lors des divers entraînements de l'étude. De tels dispositifs portables ont déjà été utilisés à l'intérieur d'études de la marche et peuvent être attachés sur différentes parties du corps, tels la hanche, la jambe ou le pied. Le Dr Handrigan suggère d'acquérir les données cinématiques au niveau de la hanche, mais il serait également possible d'obtenir des mesures à d'autres endroits. Une fois le dispositif réalisé, il sera possible d'effectuer des tests afin de déterminer quelle localisation sur le corps permet de recueillir les meilleurs résultats. Il pourrait également être possible de concevoir un système avec plus d'un capteurs (par exemple un capteur au niveau de la hanche et un autre au niveau de la jambe) pour augmenter le nombre de sources de données.

Les entraînements seront de haute intensité (94-97% de la vitesse aérobie maximale d'athlètes de haut calibre), donc il sera important que le dispositif soit fixé solidement sur l'athlète pour permettre de recueillir des données de bonne qualité.

Chaque athlète portera l'appareil pendant une durée d'au moins 1h30 (à chaque session), ce qui implique que l'autonomie de la source d'alimentation devra respecter au minimum ce temps. Toutefois, une meilleure autonomie permettrait à l'équipe de recherche d'épargner du temps si elle n'a pas besoin de recharger les piles entre chaque séance.

En termes de dimensions, le dispositif devra être le plus petit possible pour éviter de déranger les coureurs. Il devra également être protégé par un boîtier. Pour satisfaire les besoins de l'étude, le Dr Handrigan désirerait obtenir 2-3 exemplaires de l'appareil.

Les données mesurées par l'appareil seront des données cinématiques par rapport aux foulées lors de la course à pied. Pour les fins de l'étude, le Dr. Handrigan veut obligatoirement avoir accès à des données brutes. Les types de données brutes qui seront recueillies vont inclure :

- L'accélération linéaire
- L'accélération angulaire

Les données traitées qui pourraient être obtenues comprennent :

- L'identification des instants où il y a contact d'un pied avec le sol
- Fréquence des foulées et temps d'un cycle de marche
- Variabilité de la foulée

Les données devront être enregistrées pour permettre une analyse ultérieure. Le laboratoire du Dr Handrigan possède déjà un logiciel permettant d'analyser les données, mais il serait avantageux que certains traitements soient réalisés par notre équipe de projet. Un programme avec GUI (interface graphique) devra donc être conçu pour faciliter le traitement et l'analyse des données.

Chapitre 2 : Recherches et études préliminaires

2.1 Étude de faisabilité

2.1.1 Résumé des besoins

Dans l'optique d'identifier les besoins du client, les questions de base d'un projet d'ingénierie doivent être répondues. Le Tableau 1 présente les réponses à ces questions.

Tableau 1 : Réponses aux questions de base pour l'identification des besoins

Qui ?	Dr Grant Handrigan, professeur en kinésiologie, sera le client du projet.
Pour qui ?	Le dispositif sera réalisé pour les sujets de l'étude, soient 20 athlètes de cross-country de l'Université de Moncton.
Quoi ?	Concevoir un appareil pour l'acquisition de données cinétiques associées aux foulées lors de la course à pied.
Pourquoi ?	Acquérir les données nécessaires aux besoins de l'étude du Dr Handrigan.
Comment ?	À l'aide des capteurs appropriés, d'un système performant et d'une bonne analyse des données.
Où ?	Le dispositif sera porté par les athlètes à l'intérieur de la piste de cours du CEPS.
Quand?	L'appareil devra être complété avant la dernière journée de cours du semestre d'hiver (avril 2017).
Combien ?	Il faudra réaliser 2-3 appareils pour les besoins de l'étude. Le coût devra être abordable.

2.1.2 Défauts des appareils actuellement disponibles

Plusieurs produits existent déjà sur le marché afin de permettre de recueillir des données de cinématique de la marche. Par contre, plusieurs de ces produits comportent des défauts en lien avec les besoins de l'étude du Dr Handrigan. La conception d'un appareil par notre équipe pourrait possiblement régler certains de ces défauts :

- Plusieurs appareils ne permettent pas d'obtenir des données brutes, ou les rendent seulement accessibles en achetant une licence supplémentaire. Cela ne permet pas au client d'analyser les données comme il le souhaite.
- Informations limitées du vendeur par rapport au hardware. Il est difficile pour le client de bien juger la qualité des données obtenues.

- Il est parfois impossible d'apporter des modifications à l'appareil (par exemple remplacer un capteur). En cas de bris, ce qui pourrait facilement arriver lorsque l'appareil est porté pendant un exercice à intensité élevée, il serait difficile de réparer l'appareil.
- Impossibilité de changer l'emplacement du port de l'appareil sur l'athlète.
- Prix élevé, surtout pour l'achat de 2-3 appareils ou plus.

Le Tableau 2 présente une comparaison de quelques produits actuellement sur le marché. Leurs fonctionnalités et points forts sont comparés avec leurs défauts en lien avec l'étude.

Tableau 2 : Défauts de produits actuellement sur le marché

Appareil	Physilog 5 (avec GaitUp Analysis Package)	APDM Opal	Xsens MTw Awinda	Shimmer 3
Description des fonctionnalités	<ul style="list-style-type: none"> • Unité de mesure d'inertie comprenant accéléromètre et gyroscope • Accès aux données brutes • Peut être placé sur les souliers ou sur la hanche 	<ul style="list-style-type: none"> • Comprend un accéléromètre, un gyroscope et un magnétomètre • Accès aux données brutes • Peut être placé à divers endroits sur le corps • Très léger (< 25 grammes) • Autonomie de 8 heures 	<ul style="list-style-type: none"> • Comprend un accéléromètre, un gyroscope et un magnétomètre • Les données brutes ne sont peut-être pas disponibles (ou requièrent une licence) • Autonomie de 6 heures 	<ul style="list-style-type: none"> • Accéléromètre, gyroscope, magnétomètre et altimètre • Données enregistrées sur microSD ou transférées par bluetooth • Bonne autonomie (pile de 450 mAh) • Peut être désassemblé
Défauts	<ul style="list-style-type: none"> • Prix très élevé de 2200 euros pour deux appareils et une licence de logiciel pour 1 an • Faible autonomie (pile de 140 mAh) • Très difficile à changer une pièce s'il y a un bris. Il n'est pas possible d'ajouter ou de changer les capteurs 	<ul style="list-style-type: none"> • Prix probablement élevé (non disponible, sur le site Web, il faut faire une demande de <i>quote</i>) • Impossible d'apporter des modifications à l'appareil 	<ul style="list-style-type: none"> • Prix de 740 euros pour 1 seul appareil • Très difficile à changer une pièce s'il y a un bris. Il n'est pas possible d'ajouter ou de changer les capteurs 	<ul style="list-style-type: none"> • Prix de 445\$ pour un appareil • Prix de 200\$ par année pour une licence logicielle • Dimensions assez élevées pour être porté par des coureurs • Difficile d'apporter des modifications à l'appareil

2.1.3 Objectifs de travail

Les objectifs de travail pour ce projet ont été déterminés en fonction des besoins identifiés. Ces objectifs sont présentés dans la Figure 1. Les cases en bleu correspondent aux objectifs principaux, les cases en orange contiennent les objectifs spécifiques à chaque objectif principal et les cases vertes présentent les sous-objectifs.

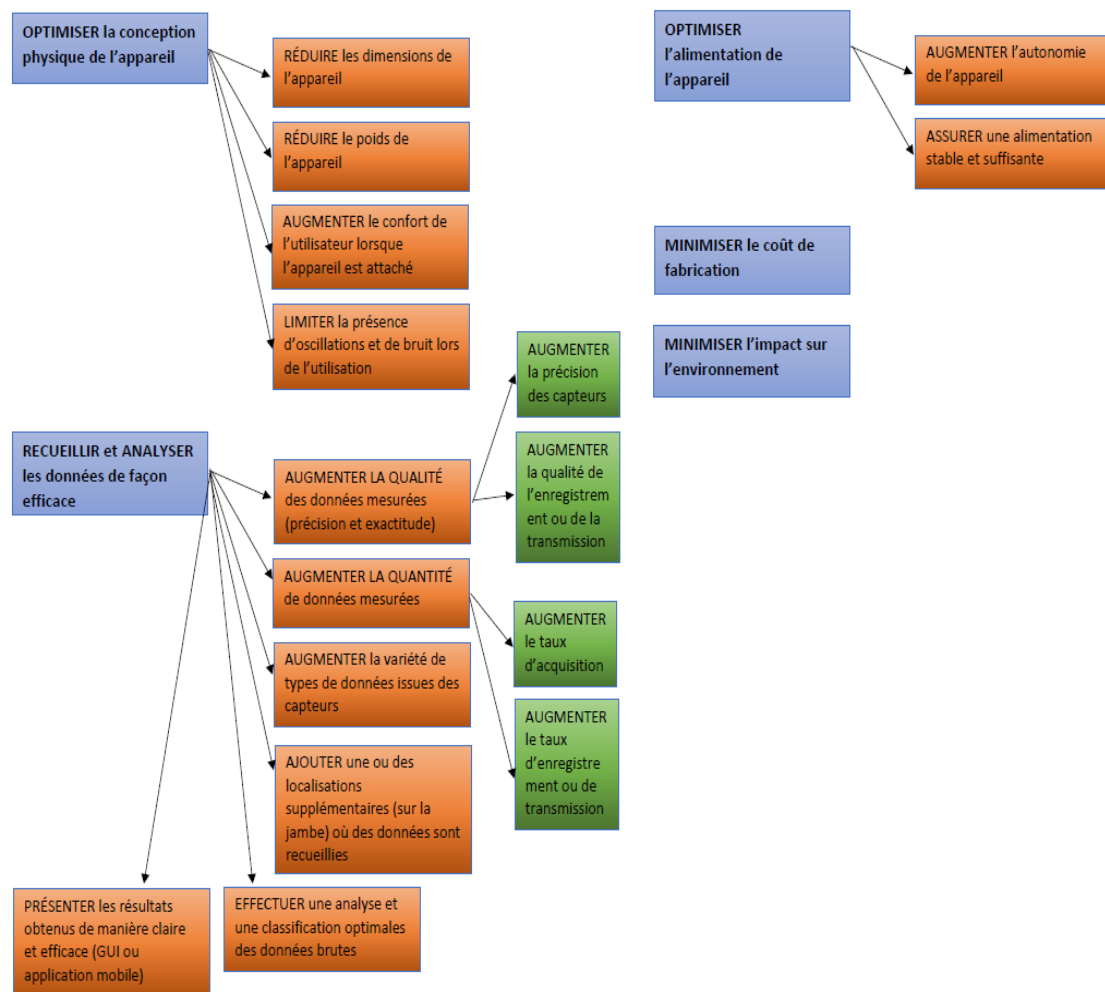


Figure 1 : Arbre des objectifs.

2.1.4 Cahier de charges

Le cahier des charges a été effectué en imposant cinq critères subdivisés en sous-critères. Le cahier de charges est démontré sur le [Tableau 3](#). Les cinq critères principaux sont présentés dans cette section.

2.1.4.1 La conception physique

La pondération pour ce critère est de 20%. Cette valeur se justifie par le fait que le produit fini étant destiné à être porté par l'athlète au niveau de la hanche, il ne faudrait pas que le matériel soit encombrant. Ce critère est à la base de tous les autres. Une mauvaise conception physique pourrait mener à des résultats erronés, car il y a de la stabilité du produit dans un sens où les capteurs utilisés sont très sensibles. Il faudrait éviter que le produit bouge en dehors des mouvements de l'athlète. Ce critère a été divisé en trois sous-critères que sont :

- **Les dimensions** (5%) : pour évaluer cet aspect, la longueur, la largeur et l'épaisseur sont mis en avant. Afin d'avoir la meilleure taille, il faudrait que : la longueur soit inférieure à 7 cm, la largeur à 4 cm et l'épaisseur à 2.5 cm.
- **Le poids** (5%) : le poids du prototype devrait être assez faible pour ne pas déranger lors de la course. Le poids optimal devrait être inférieur à 100 g.
- **Le confort** (10%) : cet aspect dépend des deux derniers sous-critères évalués précédemment. Le confort, bien que subjectif, est important pour ne pas influencer la course des participants et donc les résultats de l'étude.

2.1.4.2 Le recueil des données

Une pondération de 45% a été octroyée pour ces critères. En effet, il constitue la majeure partie du projet. C'est à ce niveau que le traitement des données et les réponses des microcontrôleurs sont étudiés pour l'analyse des résultats de l'athlète. Il y a trois sous-critères :

- **La qualité des données** (30%) : pour ce faire trois facteurs ont été mis en avant avec une pondération de 10% chaque. On a :
 - La précision des capteurs, qui doivent avoir une résolution pour une étendue de 2g ($g = 9,81 \text{ m.s}^{-2}$).
 - La stabilité du produit, en effet les capteurs présentent un certain niveau de stabilité.
 - Le bruit, il faudrait que les capteurs aient un minimum de bruit afin d'éviter que les données externes au mouvement soient recueillies. Pour une réponse optimale il faudrait un bruit inférieur à 1 LSB RMS.

- **Le nombre de types de données (10%)** : ceci permet de pouvoir avoir plusieurs types de données à étudier. Les données à analyser sont fournis par l'accéléromètre, le magnétomètre ou le gyroscope. Il faudra faire une combinaison de ces éléments. La meilleure option serait ainsi de pouvoir avoir tous les trois combinées.
- **Le nombre d'emplacements du produit (5%)** : il serait intéressant de pouvoir placer le matériel sur différents endroits sur l'athlète, par exemple la jambe, la hanche ou la cheville. L'idéal serait d'avoir trois emplacements pour pouvoir prendre des mesures.

2.1.4.3 L'alimentation de l'appareil

La pondération est de 15%. Il s'agit ici de la source d'énergie pour le produit monté sur l'athlète. Deux sous-critères ont été fixés :

- **L'autonomie de l'appareil (10%)** : il faut que la batterie soit assez puissante. Il ne faudrait pas qu'elle se décharge très rapidement. La batterie doit être assez robuste afin de pouvoir faire plusieurs essais sur une durée d'au minimum 1h30.
- **Le courant fourni par la batterie (5%)** : le courant fournit doit être au minimum de 70 mA afin de pouvoir alimenter les microcontrôleurs et les capteurs.

2.1.4.4 L'aspect environnemental

Une pondération de 5% a été fixée. Le reste des composants n'ayant pas besoin d'être changés à moins qu'ils soient défectueux, alors l'accent est mis sur la batterie. Il faudrait que ça soit rechargeable pour remplir le critère. Une batterie rechargeable permettrait d'utiliser moins de matériaux.

2.1.4.5 Le coût de fabrication

Ce critère a une pondération de 15%. Un coût minimal permettrait de réduire le budget de l'étude ou de permettre la fabrication de plusieurs appareils. Le cout optimal doit être inférieur à 150\$ par appareil.

Tableau 3 : Cahier de charges

Critère		Pond.	Barème		Min	Max
Conception physique		20%				
Dimensions						
1	a- Longueur (cm)	2%	L < 7 7 ≤ L < 9 9 ≤ L < 11 L ≥ 11	100% 75% 50% 0%		
	b- Largeur (cm)	2%	l < 4 4 ≤ l < 5 5 ≤ l < 6 l ≥ 6	100% 75% 50% 0%		
	c-Épaisseur (cm)	1%	E < 2.5 2.5 ≤ E < 3.5 3.5 ≤ E < 4.5 E ≥ 4.5	100% 75% 50% 0%		
2	Poids (en grammes)	5%	P < 100 100 ≤ P < 150 150 ≤ P < 200 200 ≤ P	100% 75% 50% 0%		200
3	Confort	10%	Très confortable Confortable Peu confortable Inconfortable	100% 75% 50% 0%		
Recueil des données		45%				
1	Qualité des données	30%				
	a- Précision des capteurs (Résolution pour une étendue de 2g d'accélération)	10%	14 bits et plus 12 bits 10 bits 8 bits et moins	100% 75% 50% 0%		
	b- Stabilité du produit	10%	Stable Légèrement stable Instable	100% 75% 50% 0%		
	c- Bruit (bruit RMS pour l'accélération)	10%	< 1 LSB rms 1-2 LSB rms 2-5 LSB rms > 5 LSB rms	100% 75% 50% 0%		
2	Nombre de types de données (Gyroscopie = Gyro., Accéléromètre = Acc., Magnétomètre = Magnéto.)	10%	Acc. + Gyro. + magnéto. Acc. + Gyro. Acc. Seul	100% 75% 50%	1 acc.	
3	Nombre d'emplacements où des mesures sont acquises	5%	n ≥ 3 n = 2 n = 1	100% 75% 50%		
Alimentation de l'appareil		15%				
1	Autonomie de l'appareil (en heures)	10%	H ≥ 5 3 ≤ H < 5 1h30min ≤ H < 3 H < 1h30min	100% 75% 50% 0%	1h 30min	
2	Courant fourni par la batterie (mA)	5%	I > 100 70 ≤ I ≤ 100 I < 70	100% 50% 0%	70	
Environnement		5%	Rechargeable Pas rechargeable	100% 0%		
Coût de fabrication (\$CAD)		15%	Coût < 150\$ 150\$ < Coût < 200\$ 200\$ < Coût < 250\$ 250\$ < Coût	100% 75% 50% 0%		

2.2 Diagramme fonctionnel

Le diagramme fonctionnel permet d'éclaircir l'ensemble du fonctionnement du système. On y trouve les liens entre les différentes fonctions du système. La Figure 2 présente le diagramme fonctionnel.

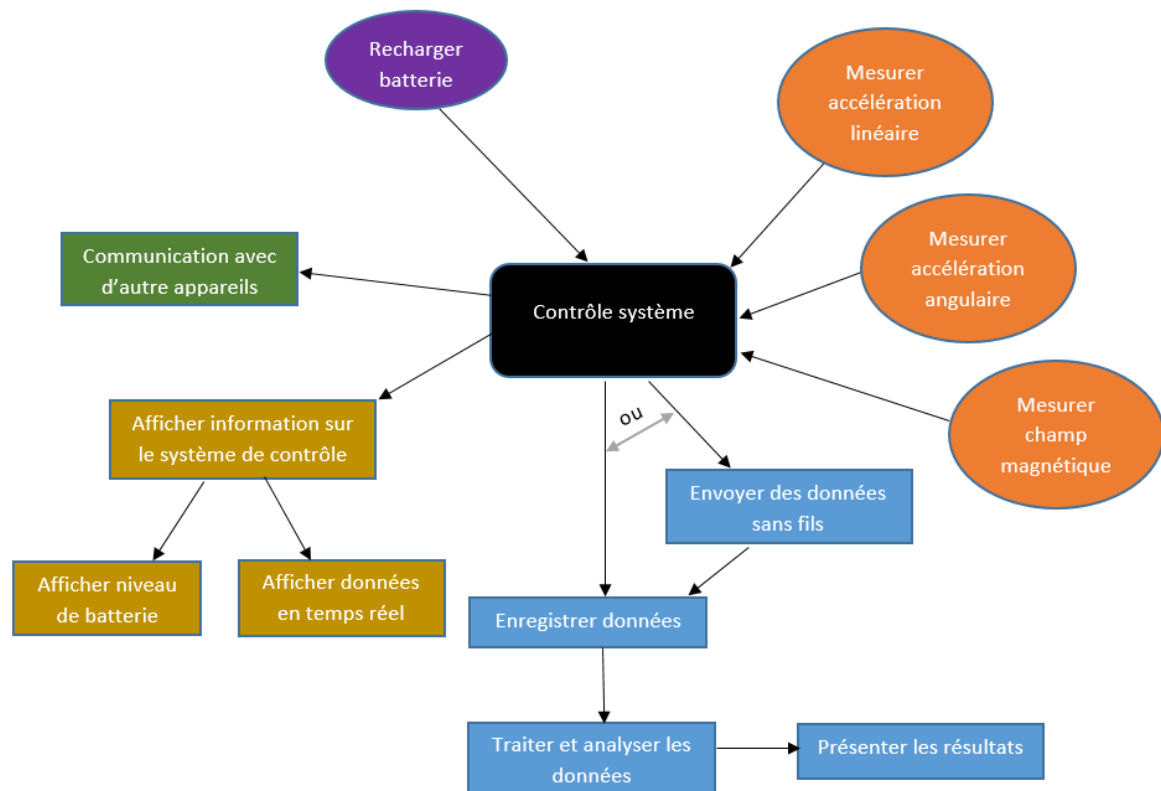


Figure 2 : Diagramme fonctionnel

2.3 Présentation des concepts

Le Tableau 4 présente les différents concepts obtenus après le brainstorming de l'équipe, à partir du diagramme fonctionnel. Ces concepts sont au nombre de douze (12) et varient l'un de l'autre par le choix du microcontrôleur, du moyen de communication, du capteur et de l'affichage. Il était systématique que la batterie soit au lithium.

Tableau 4 : Concepts proposés

Fonction	Concept 1	Concept 2	Concept 3
Microcontrôleur	Adafruit feather SD card	Conception d'un microcontrôleur	Adafruit feather SD card
Affichage	LED RGB	LED RGB	LED RGB
Batterie	Lithium	Lithium	Lithium
Communication	Micro SD	Micro SD	Micro SD
Capteur	6 DOF Gyro, Accelerometer IMU - MPU6050	FLORA Accelerometer/Compass Sensor - LSM303 - v1.0	Adafruit BNO055 Absolute Orientation Sensor
Fonction	Concept 4	Concept 5	Concept 6
Microcontrôleur	Adafruit feather SD card	Adafruit feather SD card	Adafruit feather SD card
Affichage	LED RGB	LED RGB	LED RGB
Batterie	Lithium	Lithium	Lithium
Communication	Micro SD	Micro SD	Micro SD
Capteur	SparkFun 6 Degrees of Freedom Breakout - LSM6DS3	SparkFun 9DoF Sensor Stick	SparkFun IMU Breakout - MPU-9250
Fonction	Concept 7	Concept 8	Concept 9
Microcontrôleur	Adafruit feather SD card	Sparkfun ESP32 thing	Adafruit feather SD card
Affichage	Écran LCD	LED RGB	LED RGB
Batterie	Lithium	Lithium	Lithium
Communication	Micro SD	Bluetooth/Wi-Fi	XBEE
Capteur	6 DOF Gyro, Accelerometer IMU - MPU6050	6 DOF Gyro, Accelerometer IMU - MPU6050	6 DOF Gyro, Accelerometer IMU - MPU6050
Fonction	Concept 10	Concept 11	Concept 12
Microcontrôleur	Conception d'un microcontrôleur	Adafruit feather SD card	Sparkfun ESP32 thing
Affichage	LED RGB	LED RGB	LED RGB
Batterie	Lithium	Lithium	Lithium
Communication	Micro SD	XBEE	Bluetooth/Wi-Fi
Capteur	SparkFun 6 Degrees of Freedom Breakout - LSM6DS3	Adafruit BNO055 Absolute Orientation Sensor	6 DOF Gyro, Accelerometer IMU - MPU6050

2.3.1 Microcontrôleur

Adafruit Feather SD card

Ce microcontrôleur est une carte de développement de la série Feather de Adafruit, et comme son nom l'indique, elle très petite (presque la même taille que l'Arduino Nano), légère et portable. Ce qui la rend encore plus intéressante dans le cadre de ce projet est qu'elle a un lecteur de carte SD intégré et elle a un connecteur pour batterie Li-Po et système de recharge pour la batterie. En faisant un survol rapide de ses caractéristiques, on peut retenir entre autres :

- 20 pins GPIO
- Un régulateur 3.3V avec un courant maximum de 500mA
- Une fréquence d'horloge de 8 Mhz
- Types de communication série, I2C (Inter-Integrated Circuit), et SPI (Serial Peripheral Interface)
- Un socle pour carte SD
- Un faible coût
- Supporte des batteries LiPoly 3.7/4.2V pouvant être rechargées avec micro USB
- LED pour indiquer que la batterie est faible

Sparkfun ESP32 thing

Un autre choix de microcontrôleur serait la ESP32 thing, qui est une carte de développement très récente et assez similaire à son prédécesseur, la ESP8266 du même fabricant (Sparkfun). Elle intègre sur la même carte des technologies Bluetooth, WIFI et a près de 30 pins d'entrées et sorties. Ses principales caractéristiques sont :

- 802.11 BGN WiFi intégré
- Bluetooth intégré double mode (classique et BLE)
- Plage de fonctionnement de 2,2 à 3,6 V
- 2,5 μ A de courant de sommeil profond
- Cryptage accéléré par matériel (AES, SHA2, ECC, RSA-4096)
- Mémoire flash de 4 Mo

2.3.2 Affichage

Écran LCD

Pour le mode d’affichage, un écran LCD serait un choix esthétique adéquat. Mais s’agissant d’autres aspects comme l’encombrement, le poids, la fragilité de la conception finale, ce choix devient un handicap. Les informations qui peuvent être affichées avec un tel dispositif sont : l’état de la batterie, les données brutes en temps réel et même un menu de sélection pour différents modes courses ou autres. Comme le dispositif sera placé sur un athlète, ces genres d’informations à temps réel ne seront pas très utiles.

LED RGB

Comme alternative à l’écran LCD, une LED RGB peut être utilisée. Ceci donnera des informations simples comme l’état de la batterie et s’il y a transmission ou pas. En plus, la LED RGB n’est pas aussi encombrante que la LCD.

2.3.3 Batterie

Batterie (LI-Po)

Pour le mode d’alimentation, une étude préliminaire permettant d’obtenir le meilleur type d’alimentation a été réalisée. Le Tableau 5 présente les conclusions de l’étude. Il en ressort que la batterie en lithium est de loin meilleure et la plus adaptée dans le cadre de ce projet du fait de sa légèreté, de sa durée de vie et surtout du fait qu’elle soit rechargeable. Utiliser une batterie rechargeable permet de réduire les conséquences écologiques de notre produit.

Tableau 5 : Choix de la batterie

	Plomb	Nickel-Cadmium	Nickel-Métal Hybride	Lithium
Rechargeable	Oui	Oui	Oui	Oui
Bonne durée d’ utilisation	Non	Oui	Oui	Oui
Bonne durée de vie	Non	Non	Non	Oui
Légère	Non	Oui	Oui	Oui

2.3.4 Communication

Micro-SD

Le mode de communication avec la carte micro SD permettrait de recueillir les données après la course. Il faudrait attendre que tous les tests soient effectués avant de pouvoir recueillir et analyser les données. Cette méthode ne permet pas un accès à temps réel aux données enregistrées. Ceci peut engendrer des inconvénients sur l'efficacité ou la durée de l'étude.

Sans fil (Bluetooth et Wi-Fi)

Le plus gros avantage de la communication sans fil est qu'on peut avoir accès à tout moment aux données et les analyser à partir d'un programme tiers. Contrairement au mode d'enregistrement avec la carte SD, on peut avoir un accès à temps réel aux données du coureur. Un accès continu permettrait une analyse permanente des données et donc un gain considérable en temps et en énergie pour des suggestions d'optimisation de course. L'inconvénient de ce mode de communication est qu'il faut un accès continu au point d'accès, que les deux appareils soient tout le temps connecté sinon les données seront perdues. La pertinence des données dépend de la qualité du réseau de communication et du protocole de communication.

2.3.5 Capteurs

6 DOF Gyro, Accelerometer IMU - MPU6050

Les capteurs MPU-6050 sont les premiers appareils MotionTracking du monde conçus pour les applications à faible puissance, à faible cout et à haute performance à usage portatif. Le MPU-6050 intègre la technologie MotionFusion d'InvenSense qui traite des algorithmes complexes à 6 axes. Le capteur regroupe deux modules de mesures qui sont : un accéléromètre et un gyroscope. Les caractéristiques de ce capteur peuvent être observées dans le **Erreur ! Source du renvoi introuvable..**

Tableau 6 : Caractéristiques du MPU6050

Dispositif	Code	Fonct.	Communication	Consommation	Autres
6 DOF Gyro, Accelerometer IMU - MPU6050	RB-Dfr-264	MPU6050: A-G	I2C	3,6 mA?	Digital Motion Processor (DMP) capable of processing complex 9-axis Motion Fusion algorithms
Longueur	Largeur	Épaisseur			Prix
2,1 cm	1,4 cm	?			9,90 \$ US

Module	Précision	Bruit	Zero-level	Output data rate
Accéléromètre	WL = 16 bits (+/- 2, 4, 8, 16 g) 16.384 LSB/g (±2g)	400 μ g/√Hz	±50, 80 mg	4-1000 Hz
Gyroscope	WL = 16 bits (+/- 250, 500, 1000, 2000 $^{\circ}$ /s) 131 LSB/dps (±250 dps)	0.05 $^{\circ}$ /s-rms	± 20 dps	4-8000 Hz

SparkFun 6 Degrees of Freedom Breakout - LSM6DS3

Ce capteur a un accéléromètre numérique 3D et un gyroscope 3D et même un capteur de température embarqué et le tout fonctionnant à faible puissance. Le capteur peut fonctionner jusqu'à une fréquence de 1,6 kHz, entraînant une légère augmentation de la puissance consommée. Les caractéristiques de ce capteur peuvent être observées sur le Tableau 7.

Tableau 7 : Caractéristiques du LSM6DS3

Dispositif	Code	Fonct.	Communication	Consommation	Autres
SparkFun 6 Degrees of Freedom Breakout - LSM6DS3	SEN-13339 ROHS	LSM6DS3: A-G	I2C ou SPI	0,9 mA	Temperature Pedometer, step detector and step counter
Longueur	Largeur	Épaisseur			Prix
2 cm	2 cm	~1 mm			9,95 \$ US

Module	Précision	Bruit	Zero-level	Output data rate
Accéléromètre	WL = 16 bits* (+/- 2, 4, 8, 16 g)	90 μ g/√Hz	±40 mg	13 – 6664 Hz
Gyroscope	WL = 16 bits* (+/- 125, 245, 500, 2000 $^{\circ}$ /s)	7 mdps/√Hz	± 10 dps	13 – 1666 Hz

SparkFun 9DoF Sensor Stick

C'est une barre de capteurs intégrant plusieurs capteurs : l'accéléromètre ADXL345, le magnétomètre HMC5883L et le gyroscope ITG-3200 MEMS. La « barre » communique via une interface I2C simple. Comparativement aux autres capteurs, elle est plus complète vu qu'elle intègre trois (3) capteurs de technologie avancée et utilise une communication assez simple pour la transmission des données. Les caractéristiques de ce capteur peuvent être observées sur le tableau 8.

Tableau 8 : Caractéristiques du 9DoF Sensor Stick

Dispositif	Code	Fonct.	Communication	Consommation	Autres
SparkFun 9DoF Sensor Stick	SEN-13944 ROHS	LSM9DS1: A-G-M	I2C ou SPI	4,6 mA 3,3 V	The biggest difference between the two boards, besides the slimmed down 0.9"x0.4" footprint, is the number of broken-out pins, with the Sensor Stick featuring only four for quick setup and ease of use.
Longueur	Largeur	Épaisseur			Prix
2,5 cm	1 cm	~1 mm			14,95 \$ US

Module	Précision	Bruit	Zero-level	Output data rate
Accéléromètre	WL = 16 bits (+/- 2, 4, 8, 16 g)		±90 mg	
Gyroscope	WL = 16 bits (+/- 245, 500, 2000 °/s)		± 30 dps	
Magnétomètre	WL = 16 bits (+/- 4, 8, 12, 16 Gauss)		± 1 G	

2.4 Étude de faisabilité

Des douze (12) concepts recueillis après le « brainstorming », il n'en reste que cinq après l'étude de faisabilité (Tableau 9). Pour optimiser le tri et choisir le meilleur design qui respecterait la majeure partie des points du cahiers de charges, les quatre concepts qui présentent une différence majeure pour le choix des composants, soient les concepts 1, 4, 5 et 12, passent à une étape de pondération à travers la matrice de décision. Le concept obtenant la plus grande pondération sera retenu comme concept final à réaliser.

Tableau 9 : Analyse des concepts

Concepts	Physique	Économique	Temporel	Socio-Env.	Décision
Concept 1	Oui	Oui	Oui	Oui	Retenu
Concept 2	Oui	Oui	Non	Oui	Rejeté
Concept 3	Oui	Non	Oui	Oui	Rejeté
Concept 4	Oui	Oui	Oui	Oui	Retenu
Concept 5	Oui	Oui	Oui	Oui	Retenu
Concept 6	Oui	Oui	Oui	Oui	Retenu
Concept 7	Non mais	Oui	Oui	Oui	Retenu mais
Concept 8	Oui mais	Oui	Oui	Oui	Retenu mais
Concept 9	Oui	Oui mais	Oui	Oui	Retenu mais
Concept 10	Oui	Oui	Non	Oui	Rejeté
Concept 11	Oui	Non	Oui	Oui	Rejeté
Concept 12	Oui mais	Oui	Oui	Oui	Retenu

2.4 Matrice de décision

La matrice de décision permet de faire le choix final parmi les quatre concepts restants. Elle est divisée en 5 parties principales, soit la conception physique, le recueil des données, l'alimentation de l'appareil, l'environnement et le coût de fabrication. À chaque partie et sous-partie est attribué un pourcentage qui reflète l'importance de cette dernière. La matrice de décision peut être observée dans le

Tableau 10.

- La conception physique est à 20% avec 3 sous-critères : les dimensions, le poids et le confort. Cette partie est importante, car l'appareil doit être le moins dérangerant possible pour ne pas biaiser les performances de l'athlète.
- Le recueil des données est un critère très important et son barème est donc de 40%. Les sous-critères sont la qualité des données (précision des capteurs, décalage à zéro et le bruit RMS pour l'accélération) et le nombre de types de données acquis par l'appareil. Les données doivent être de qualité pour assurer les meilleurs résultats possibles des études effectuées à l'aide de l'appareil. Le nombre de donnée est important, car il augmente les options qui sont offertes à l'utilisateur.
- L'alimentation de l'appareil est bien sûr un autre critère à considérer, avec une valeur de 15%. L'appareil doit avoir une grande autonomie d'énergie pour pouvoir être le plus autonome possible. L'utilisateur devrait pouvoir se servir de l'appareil sur de longues périodes.

- La conception de l'appareil doit également prendre compte du respect de l'environnement qui se voit attribuer la pondération de 5%. Les composantes choisies doivent respecter la norme RoHS (Restriction de Substances Dangereuse, traduit de "Restriction of Hazardous Substance") et la batterie sélectionnée doit aussi être rechargeable afin d'éviter le gaspillage.
- Finalement, le dernier critère est le coût de fabrication en dollar canadien. Celui-ci doit être raisonnable et mérite une part de 20% dans la conception, étant donné que c'est un critère important pour le client.
- Après évaluation des quatre concepts proposés en fonction du cahier des charges, le concept gagnant est le concept 4, qui répond le plus aux besoins exigés par le client, avec une pondération de 93% des critères. **L'appareil sera donc constitué d'un microcontrôleur Sparkfun ESP32 thing, d'un affichage à LED RGB, d'une batterie au lithium, d'une communication sans-fil Bluetooth ou Wi-Fi et d'un capteur 6 Degrees of Freedom MPU6050 (accéléromètre et gyroscope).**

Tableau 10 : Matrice de décision

Critère		Pond.	Concept 1	Concept 4	Concept 5	Concept 12
Conception physique		20%				
1	Dimensions	5%	5%	5%	5%	5%
2	Poids (en grammes)	5%	5%	5%	5%	5%
3	Confort	10%	10%	10%	10%	10%
Recueil des données		40%				
1	Qualité des données	30%				
	a- Précision des capteurs (Résolution pour une étendue de 2g d'accélération)	10%	10%	10%	10%	10%
	b- Décalage à zéro (accéléromètre)	10%	7.5%	10%	5%	7.5%
	c- Bruit (bruit RMS pour l'accélération)	10%	7.5%	10%	5%	7.5%
2	Nombre de types de données (Gyroscope = Gyro., Accéléromètre = Acc., Magnétomètre = Magnéto.)	10%	7.5%	7.5%	10%	8%
Alimentation de l'appareil		15%				
1	Autonomie de l'appareil (en heures)	10%	5.0%	2.5%	5.0%	10%
2	Courant fourni par la batterie (mA)	5%	5%	5%	5%	5%
Environnement		5%	5%	5%	5%	5%
Coût de fabrication (\$CAD)		20%	20%	20%	10%	20%
Total		100%	88%	90%	75%	93%

2.4.1 Explication

Le Tableau 10 démontre que les trois capteurs possibles pour le système d'acquisition des données sont l'accéléromètre, le gyroscope et le magnétomètre. De ces trois capteurs, seulement l'accéléromètre est essentiel pour le client mais les deux autres sont des ajouts très appréciés.

Dans le concept final choisi à l'aide de la matrice de décision, un module incluant un accéléromètre et un gyroscope a été retenu en grande partie dû au fait que la qualité des données en fonction du prix étaient meilleurs pour le module contenant simplement ces deux types de capteurs. Il faut également noter qu'un module contenant les trois capteurs a été acheté pour mener des tests et possiblement faire la conception d'un système un peu plus sophistiqué pour aider les recherches de M. Handrigan.

Le traitement des données a été ajouté aux tâches afin d'ajouter plus de contenu au projet. Un microcontrôleur avec émetteur Bluetooth et WI-FI a été acheté pour tester pour le mode de transfert de données sans fil et sera implanté dans le système pour l'acquisition des données. L'affichage du niveau de batterie sera fait à l'aide d'une diode RGB et l'affichage des données en temps réels sera fait sur une interface graphique. Il n'y aura pas de communication entre plusieurs appareils puisqu'un seul appareil peut recueillir la totalité des données nécessaire pour l'étude. Pour l'alimentation, une batterie au lithium sera utilisée et pourra être rechargée à l'aide d'une broche micro USB.

2.5 Échéancier

L'échéancier, créé avec le logiciel Microsoft Project, comprend la totalité du travail à accomplir par l'équipe. Durant le déroulement du projet quelques tâches n'ont pas été accomplies à temps. Ces retards étaient causés par des imprévus comme le mal fonctionnement d'un logiciel ou des complications durant la livraison des composantes. Par contre, elles ont toutes été accomplies avec succès. Le Tableau 11 démontre l'échéancier mis en place durant la partie initiale du projet.

Tableau 11 : Échéancier initial du projet

Nom de la tâche	Durée	Début	Fin	Prédi	Noms ressources
Commande des pièces	2 jours	Lun 2/27/17	Mar 2/28/17		Aboubacar,Andrée-Maude,François,Khadija,Samuel
➤ Contrôle du système	19.86 jours	Lun 3/6/17	Ven 3/31/17		François,Khadija
Programmation pour microcontrôleur	20 jours	Lun 3/6/17	Ven 3/31/17		François,Khadija
Affichage de l'état de la batterie	20 jours	Lun 3/6/17	Ven 3/31/17		François,Khadija
➤ Mesure des données	14 jours	Mar 2/28/17	Ven 3/17/17		Aboubacar,Samuel
Mise en fonction des capteurs	7 jours	Mar 2/28/17	Mer 3/8/17		Samuel,François,Aboubacar
Capteurs fonctionnels	0 jour	Jeu 3/9/17	Jeu 3/9/17	6	
Mise en fonction de l'algorithme de fusion	5 jours	Jeu 3/9/17	Mer 3/15/17	6	Samuel,Aboubacar,François
Configuration optimale des capteurs (précision, etc.)	7 jours	Jeu 3/9/17	Ven 3/17/17	6	Aboubacar,Samuel
➤ Enregistrement et transmission des données	18 jours	Mar 3/7/17	<u>Jeu 3/30/17</u>		Andrée-Maude,Khadija
MicroSD	7 jours	Jeu 3/9/17	Ven 3/17/17	6	Andrée-Maude,Khadija
Bluetooth	14 jours	Mar 3/7/17	Ven 3/24/17		Andrée-Maude,Khadija
Essai d'un récepteur bluetooth et calcul de la portée	4 jours	Ven 3/24/17	Jeu 3/30/17	12	Andrée-Maude,Khadija
➤ Conception du boîtier	19 jours	Lun 2/27/17	Jeu 3/23/17		Andrée-Maude,Samuel
Modèle 3D	16 jours	Lun 2/27/17	Lun 3/20/17		Samuel
Attache	16 jours	Lun 2/27/17	Lun 3/20/17		Andrée-Maude
Minimiser les vibrations	3 jours	Mar 3/21/17	Jeu 3/23/17	15,16	Andrée-Maude,Samuel
➤ Analyse des données	25 jours	Lun 2/27/17	Lun 4/3/17		Aboubacar,François
Interface graphique	5 jours	Lun 3/27/17	Ven 3/31/17	20	François
Programme Matlab	20 jours	Lun 2/27/17	Ven 3/24/17		Aboubacar
Fin de la présentation des données	0 jour	Lun 4/3/17	Lun 4/3/17	19	
Assemblage (mise en commun)	2 jours	Ven 3/31/17	Mar 4/4/17	3,9,13,1	Aboubacar,Andrée-Maude,François,Khadija,Samuel
Test final du produit	2 jours	Mar 4/4/17	Jeu 4/6/17	22	Aboubacar,Andrée-Maude,François,Khadija,Samuel
Rapport final	5 jours	Mar 4/4/17	<u>Mar 4/11/17</u>	22	Aboubacar,Andrée-Maude,François,Khadija,Samuel
Remise du rapport	0 jour	Mar 4/11/17	Mar 4/11/17	24	

Chapitre 3 : Design de l'appareil

3.1. Circuit intégré

La conception du circuit imprimé a été réalisée avec le logiciel Eagle. Il est composé principalement du microcontrôleur, du capteur de données, de la batterie, de l'interrupteur ainsi que de la diode RGB. Trois résistances sont placées en parallèles pour contrôler la diode RGB et deux autres sont placées en séries pour créer un diviseur de tension afin de mesurer le niveau de tension dans la batterie. La Figure 4 démontre les connexions effectuées entre chaque composante.

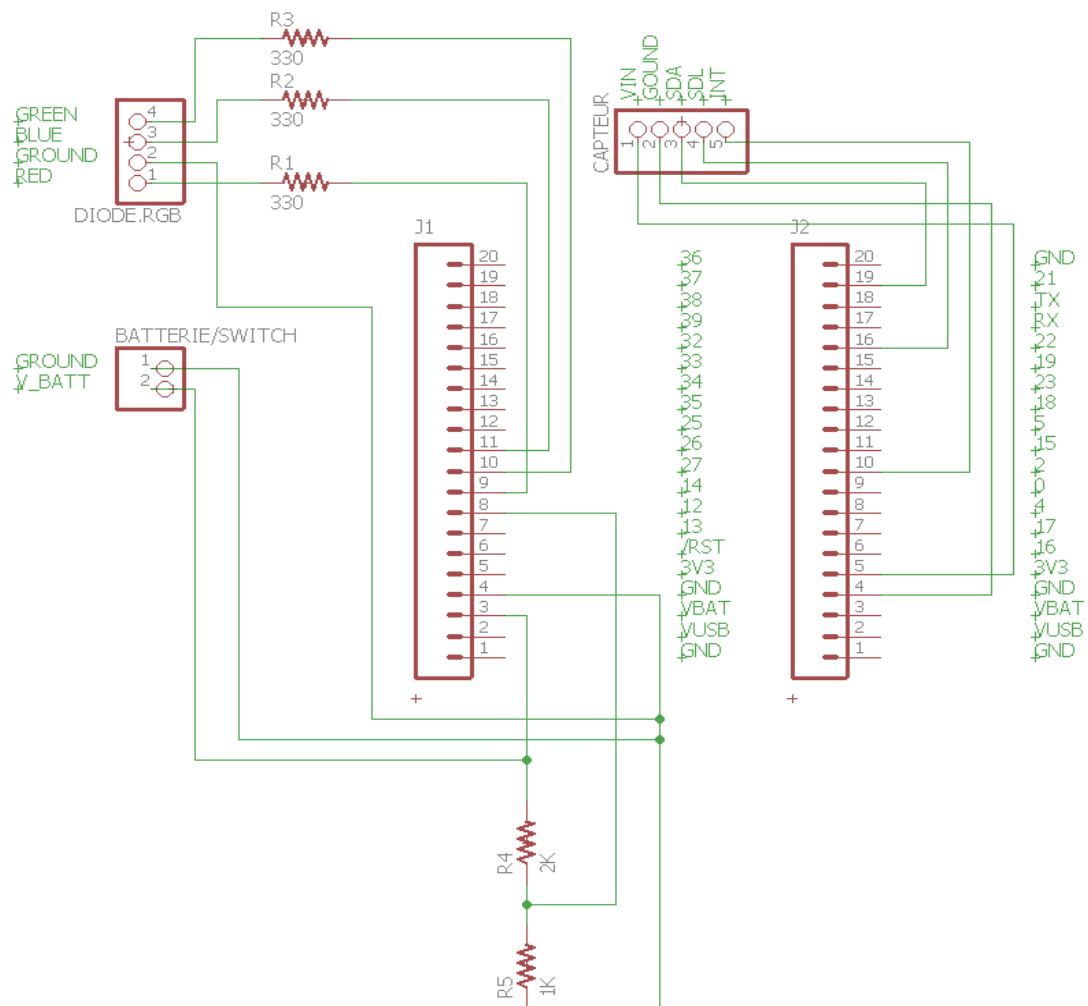


Figure 4 : Connexion sur le circuit imprimé

La caractéristique la plus importante du circuit imprimé est la dimension puisque l'appareil de collecte de données doit être le plus petit possible pour accommoder les athlètes. Avec cette exigence en tête, un circuit mesurant 8cm x 4.3cm a été conçu. La conception du circuit imprimé est bénéfique pour plusieurs raisons. Premièrement, le circuit imprimé permet de maintenir facilement chaque composante en place dans le boîtier. Deuxièmement, avec le circuit imprimé, il est possible de connecter la batterie au microcontrôleur en passant par les broches V_{in} et GND au lieu de passer par la prise conçue pour brancher la batterie. Ce mode de branchement est avantageux puisque le branchement de la batterie par la prise encombre l'espace de recharge de la batterie qui est situé à proximité. Troisièmement, le circuit imprimé donne accès aux dimensions nécessaires pour faire la conception du boîtier à l'aide de SolidWorks. La Figure 5 démontre le circuit imprimé fait à l'aide du logiciel Eagle.

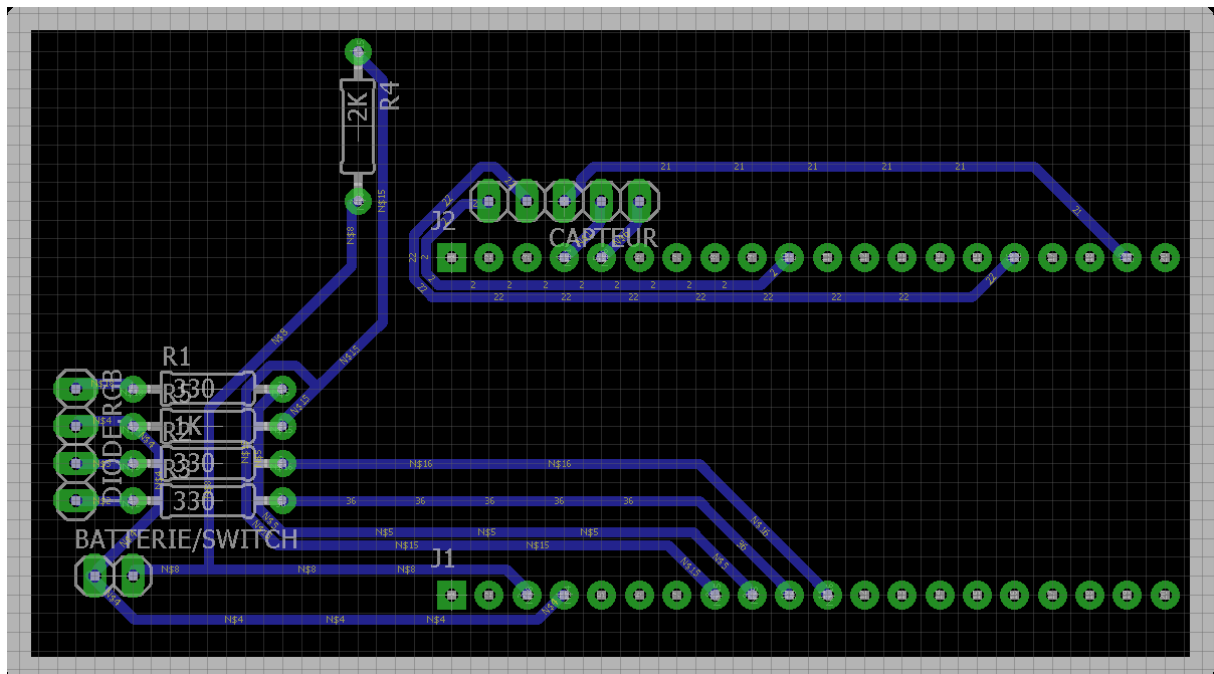


Figure 5 : Design final du circuit imprimé

3.2 Boîtier

La conception du boîtier dans lequel sera assemblé le montage final est fait à l'aide du logiciel SolidWorks. Les dimensions ont été spécifiquement choisies à l'aide des dimensions de toutes les composantes formant l'appareil. Les dimensions finales externe du boîtier sont de 8,52 cm de longueur, 6,10cm de largeur et 2,80 cm de profondeur. Les différentes vues sont démontrées sur la Figure 5. Sur la vue de face et de derrière, on peut apercevoir les trous pour passer la ceinture qui va servir d'attache sur l'athlète. Sur la vue de côté gauche, le trou rectangulaire est l'emplacement de l'interrupteur tandis que le trou circulaire est l'emplacement de la diode. Sur la vue de côté droit, on peut remarquer le trou qui permettra à l'utilisateur de brancher l'appareil à l'aide d'un chargeur micro USB. La Figure 6 représente le résultat final du boîtier en trois dimensions.

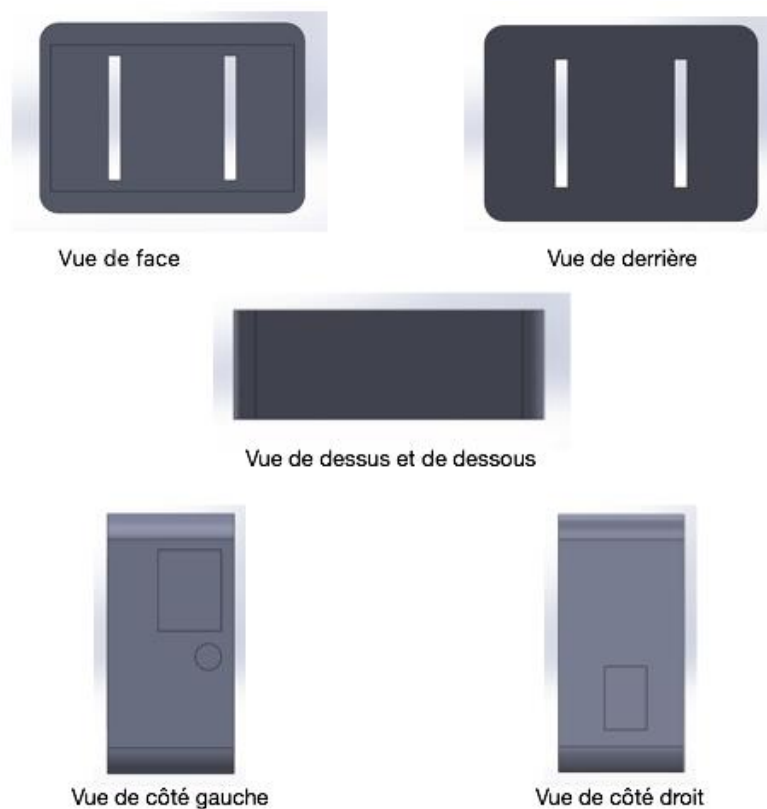


Figure 5 : Différentes vues du boîtier

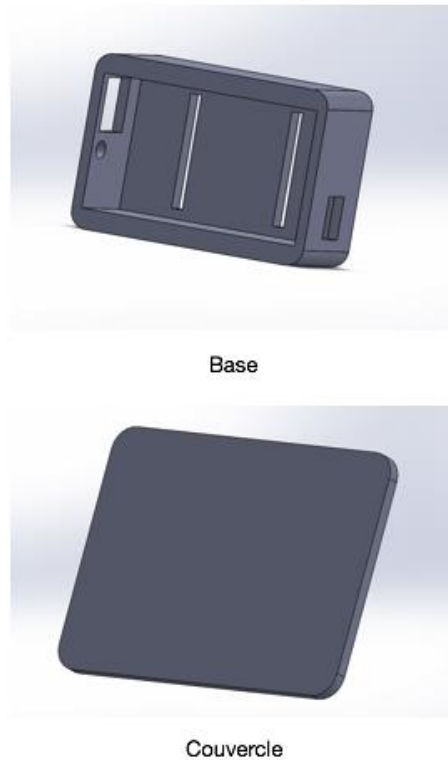


Figure 6 : Conception finale 3D du boîtier

3.3 Développement du logiciel de système embarqué

Le code sur le système embarqué a pour fonctions :

- Lire les données de l'accéléromètre et du gyroscope
- Organiser les mesures prises dans un tableau pour un envoi en bloc
- Convertir les valeurs en bytes pour l'envoi sur le réseau
- Initialiser la communication UDP
- Attendre une requête pour effectuer le transfert des données
- Afficher l'état de la batterie en temps réel

Le développement des codes du système embarqué peut se faire à l'aide de deux environnements de développement qui sont : l'environnement de développement Arduino et ESP-IDF (Espressif IoT Development Framework) qui est l'environnement de développement natif des cartes ESP.

Pour la programmation de l'ESP à l'aide de l'IDE d'Arduino, il faut ajouter un Add-on qui ajoute une liste complète des cartes programmables ESP 32 sur l'IDE. L'installation

du complément ESP est assez simple et est expliquée en détail par le projet GitHub [3] « [arduino-esp32/doc/windows.md](https://github.com/espressif/arduino-esp32/blob/master/doc/windows.md) »¹ disponible gratuitement. Bien que l'environnement de développement IDE Arduino soit très populaire, les ressources et tutoriels sont moins garnis que ceux disponibles sur l'ESP-IDF.

ESP-IDF est l'environnement de développement natif des cartes embarquées ESP, elle est gratuite, par défaut elle est utilisée avec la console de commande. Il est préférable de l'utiliser avec une des distributions Linux. Pour plus de commodité, elle peut être intégrée à un IDE de développement comme Eclipse. Il existe un site² dédié à l'ESP-IDF [4] (guide d'installation, bibliothèques et exemples).

Tout le code allant sur la carte programmable a été effectué sur l'IDE d'Arduino parce que la plateforme est pratique et que tous les membres du groupe y sont familiers. La Figure 7 donne l'ordinogramme de la partie capteurs et communication du code.

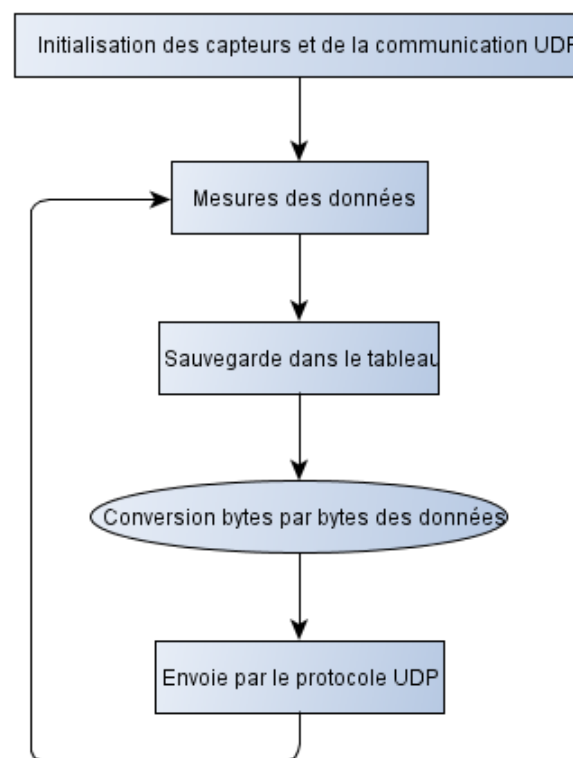


Figure 7 : Ordinogramme du code de système embarqué

¹ Lien : <https://github.com/espressif/arduino-esp32/blob/master/doc/windows.md>

² Lien : <https://esp-idf.readthedocs.io/en/latest/windows-setup.html>

3.3.1 Mesures d'accélération et gyroscope

Le code écrit pour le microprocesseur (EnvoiUDPArduino.ino) peut être trouvé à l'adresse suivante :

<https://github.com/efl7126/ProjetGELE3700>.

Pour les mesures de données à partir du capteur, une bibliothèque nommée MPU6050.h permet d'extraire facilement les données. À partir de la bibliothèque, on peut avoir accès soit aux valeurs « raws » (brutes) ou aux valeurs normalisées. Le capteur permet d'avoir plusieurs échelles de mesure que se soit pour le gyroscope ou pour l'accéléromètre avec chacun la sensibilité correspondante et le niveau d'erreur.

Tableau 12 : Calibres du capteur MPU-6050 (gyroscope et accéléromètre)

Gyro Full Scale Range (°/sec)	Gyro Sensitivity (LSB/°/sec)	Gyro Rate Noise dps/VHz	Accel Full Scale Range (g)	Accel Sensitivity LSB/g
±250	131	0.005	±2	16384
±500	65.5	0.005	±4	8192
±1000	32.8	0.005	±8	4096
±2000	16.4	0.005	±16	2048

Dans cette étude, les calibres ±2000 °/sec (gyroscope) et ±2 g (accélération) sont utilisés.

Pour recueillir les valeurs brutes de l'accéléromètre et les stocker dans un objet de classe 'Vector', on utilise la fonction suivante :

```
Vector rawAccel = mpu.readRawAccel();
```

Pour les valeurs normalisées, on utilise celle-ci :

```
Vector normAccel = mpu.readNormalizeAccel();
```

Pour le gyroscope, on a :

- Valeurs brutes : `Vector rawGyro = mpu.readRawGyro();`
- Valeurs normalisées : `Vector normGyro = mpu.readNormalizeGyro();`

L'exemple ci-dessous montre comment faire la sauvegarde des valeurs fournies par l'accéléromètre :

```
ts_BufferDonnees.valeursX[indiceTableau] = rawAccel.XAxis;
ts_BufferDonnees.valeursY[indiceTableau] = rawAccel.YAxis;
ts_BufferDonnees.valeursZ[indiceTableau] = rawAccel.ZAxis;
```

3.3.2 Envoi UDP

L'envoi des données est la partie la plus cruciale du code du système embarqué car il faut qu'on puisse envoyer les données avec le bon format à travers le protocole de communication sans fil. Ici, le protocole de communication choisi est UDP (protocole de datagramme utilisateur) qui est l'un des principaux protocoles de télécommunication utilisés par internet. Le protocole UDP permet la transmission de données de manière très simple entre deux entités, chacune étant définie par une adresse IP et un numéro de port. Il permet la transmission rapide de petites quantités de données depuis un serveur vers de nombreux clients. Dans le cas de ce projet, le microcontrôleur sera le serveur et tous les appareils qui se connecteront dessus pour recevoir les données des capteurs seront les clients.

L'envoi à travers le canal UDP se fait par paquet d'octets, donc avant de transmettre des valeurs de types 'float', il faut les convertir en 4 octets.

Après l'enregistrement des données dans le tableau, on les convertit en octets et on les agence de la manière suivante dans le tableau. Chaque valeur prend quatre octets à cause de leur type 'float'. Au totale, on peut envoyer un tableau de 128 octets, donc 32 valeurs à la fois :

- 16 valeurs pour chaque axe de l'accéléromètre
- 16 valeurs pour chaque axe du gyroscope
- Et 32 marqueurs pour chaque lot de valeurs

Le

Tableau 13 montre une représentation graphique du format d'envoi des données.

Tableau 13 : Trame d'envoi des données

Marqueur Accel	Accel.X	Accel.Y	Accel.Z	Marqueur Gyro	Gyro.X	Gyro.Y	Gyro.Z	Marqueur Accel	Accel.X	...
-------------------	---------	---------	---------	------------------	--------	--------	--------	-------------------	---------	-----

Dans le code, le marqueur pour l'accéléromètre et le gyroscope on la même valeur, soit 9999.

Le code fonctionne de telle sorte qu'à chaque fois qu'on reçoit une requête du client, les valeurs contenues dans le tableau lui sont envoyées automatiquement. La requête dans ce cas est '55'. Donc, pour une réception continue des valeurs, le client doit envoyer continuellement la requête.

Pour plus de détails sur le code, le fichier source³ est disponible sur le GitHub du groupe.

3.3.3 Alimentation et LED

L'affichage du niveau de la batterie est un aspect important, dans un sens où il permet de savoir si le produit est utilisable à l'instant ou pas. Le code rédigé avec le logiciel Arduino IDE permet de connaître le niveau de la batterie et de gérer l'affichage avec la LED RGB.

La batterie utilisée est le PKCELL LP503562. Elle a une capacité de 1200mAh et une tension nominale de 3.75V. La plage de fonctionnement est entre 2.75V (tension de coupure de décharge) et 4.2V (tension de coupure de charge)⁴[2]. Le microcontrôleur a une plage de fonctionnement entre 2.2V et 3.6V.

Afin de pouvoir afficher le niveau de la batterie, la méthode utilisée est le diviseur de tension. Cette méthode a été utilisée car les pins I/O ne prennent que 3.3V maximum. Il faudrait alors réguler la tension qu'on met à leur borne. Deux résistances ont été utilisées : une de 20 kiloOhms et une autre de 10 kiloOhms. La tension d'entrée de la batterie est mesurée à partir de la pin **VBAT** du microcontrôleur. Cette dernière est reliée à la **pin 12** du microcontrôleur à travers le circuit de diviseur de tension.

Pour faire le branchement de la LED, trois résistances de 330 ohms ont été utilisées. Ces résistances permettent de diminuer le courant arrivant aux pins RGB de la LED. La Figure 8 montre le circuit de branchement, la Figure 9 montre la plage de pourcentages des tensions avec la couleur correspondante et la Figure 10 montre l'organigramme du code [1] réalisé sur Arduino IDE pour l'affichage de l'état de la batterie.

³ Lien : https://github.com/efl7126/ProjetGELE3700/blob/Arduino/EnvoiUDPArduino_3.ino

⁴ Voir annexe pour spécification de la batterie.

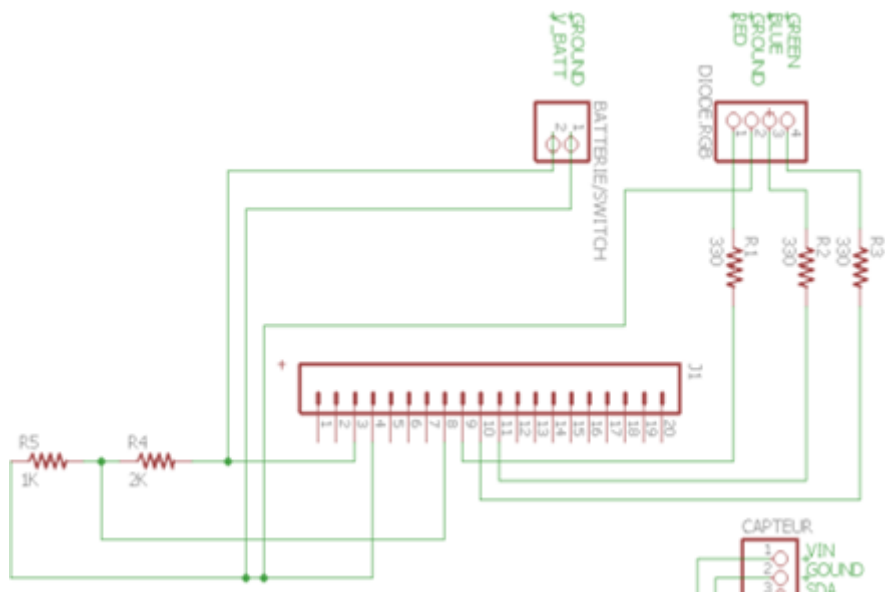


Figure 8: Montage du circuit de la batterie et de la LED.

0%	2.75
10%	2.895
20%	3.04
30%	3.185
40%	3.33
50%	3.475
60%	3.62
70%	3.765
80%	3.91
90%	4.055
100%	4.2

Figure 9 : Couleur de la LED en fonction du niveau de tension

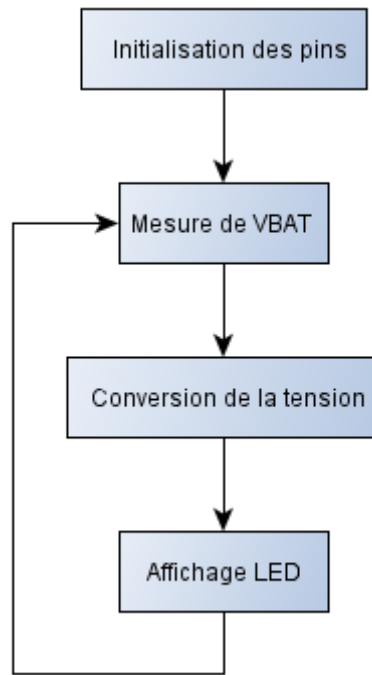


Figure 10 Organigramme du programme d’affichage d’état de la batterie

Chapitre 4 : Développement du logiciel pour le recueil et l'analyse des données

Le développement logiciel pour le recueil et l'analyse des données est l'une des composantes essentielles sur lequel le groupe a dû travailler durant le projet. En effet, les données envoyées par les capteurs (accéléromètre et gyroscope) doivent être enregistrées, traitées, et ensuite analysées. Toutes ces fonctions doivent être implémentées à l'aide de codes de programmation. La

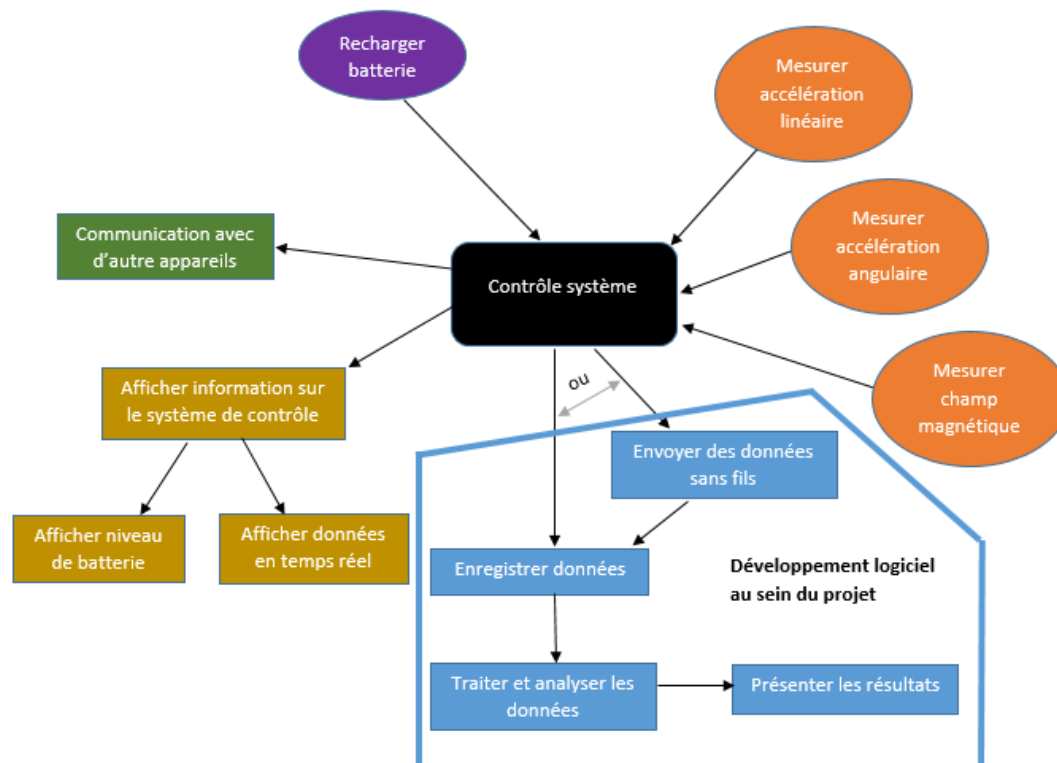


Figure 11 reprend le diagramme fonctionnel du projet, présenté plus tôt dans ce rapport, et montre les fonctions de l'appareil qui sont reliées au développement logiciel.

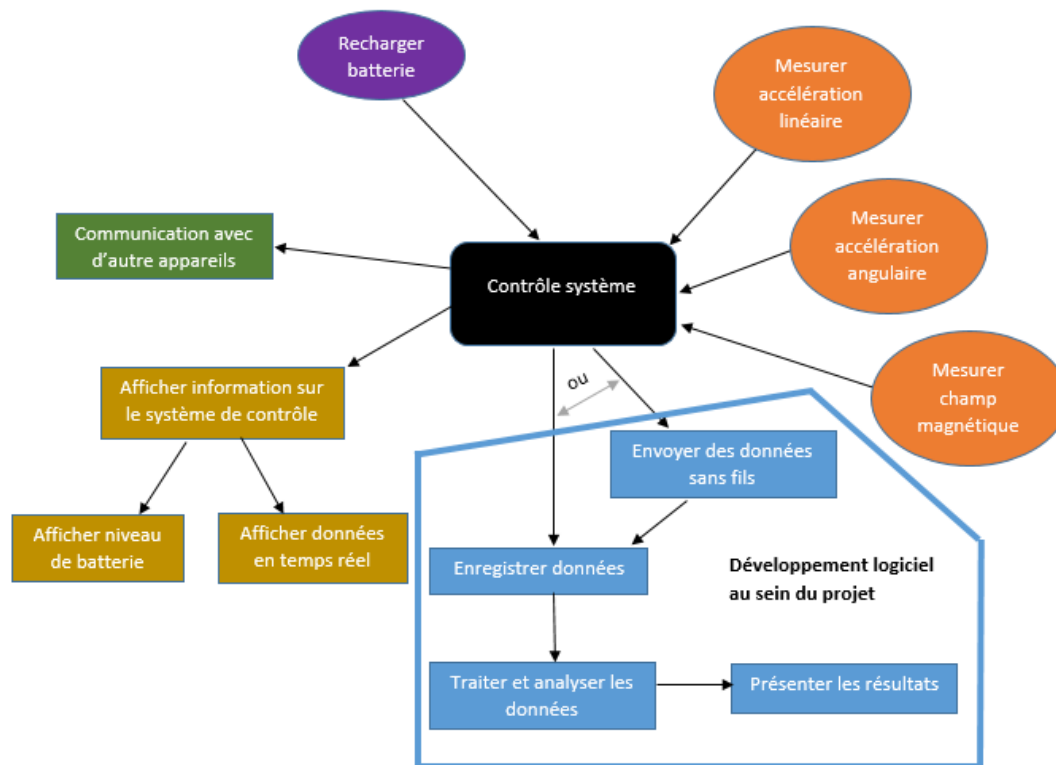


Figure 11 : Diagramme fonctionnel avec encadré des fonctions liées au développement logiciel

4.1 Architecture logicielle du projet

Pour mieux comprendre les fonctionnalités liées à l'envoi, à l'enregistrement, au traitement et à l'analyse des données, un schéma bloc des composantes logicielles du projet est présenté à la Figure 12.

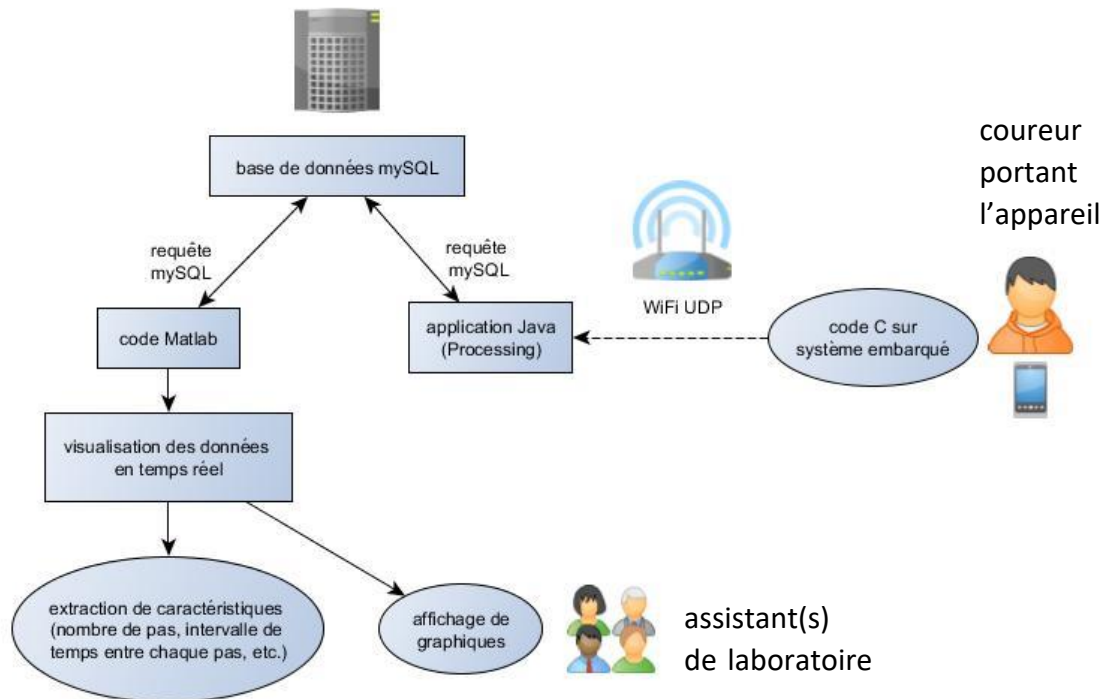


Figure 12 : Schéma bloc des composantes logicielles

Le schéma commence à l'extrémité droite avec un coureur participant à l'étude qui porte l'appareil sur lui. Un code en C pour système embarqué est programmé à l'intérieur du microcontrôleur de l'appareil (voir section 3.3). Ce code permet de recueillir les données du capteur MPU-6050 (accéléromètre et gyroscope), correspondant respectivement aux accélérations linéaires et angulaires.

Ces données sont ensuite transmises par communication sans-fil sur le réseau WiFi de l'université (umcm-projet). Le protocole utilisé pour la transmission des données est le protocole UDP. Une autre option qui pourrait être ajoutée sur l'appareil serait de recueillir les données avec une carte microSD au lieu de les envoyer par communication sans fil.

Un programme écrit en Java et utilisant les bibliothèques Processing permet de recueillir les données envoyées sur le port UDP. Ce code a deux fonctions : recueillir les données et ensuite les envoyer vers une base de données mySQL par l'entremise d'une requête. La base de données est très importante afin d'enregistrer et de classer les données recueillies par les capteurs.

Les données entreposées dans la base de données mySQL peuvent ensuite être analysées à partir d'un programme écrit en Matlab. Ce programme réussit à accéder à la base de données à travers de requêtes mySQL. L'objectif du programme Matlab est de permettre une visualisation des données recueillies par l'appareil en utilisant une GUI (interface graphique). Les données peuvent être visualisées sous forme de graphiques en temps presque réel.

Finalement, le code Matlab permet aussi d'analyser les données. Des caractéristiques reliées aux pics (nombre de pas, temps moyen entre les pas, prominence des pas) sont extraites et présentées dans la GUI. Un autre rôle du programme Matlab est de produire des fichiers de données brutes sous format .csv pour que les assistants du laboratoire de Grant Handrigan puissent analyser les données à leur guise.

4.2 Gestion du développement avec GitHub

La plateforme de gestion de développement logiciel GitHub a été utilisée dans ce projet. Cette plateforme s'est avérée très importante pour que tous les membres de l'équipe puissent travailler sur plusieurs fichiers. Il devient ainsi facile de voir les modifications apportées aux codes. Par exemple, si quelqu'un apporte une modification qui cause un bogue, il est facile de retourner en arrière et de trouver la cause. GitHub permet également de bien organiser les différents codes du projet.

Le dépôt (*repository*) des codes du projet est disponible en ligne en suivant le lien suivant :

→ <https://github.com/efl7126/ProjetGELE3700/>

Le dépôt est accessible au public, c'est-à-dire qu'il n'est pas protégé par un mot de passe. La Figure 13 montre un aperçu de la plateforme GitHub lorsque le projet était encore en phase de développement (le 2 avril 2017).

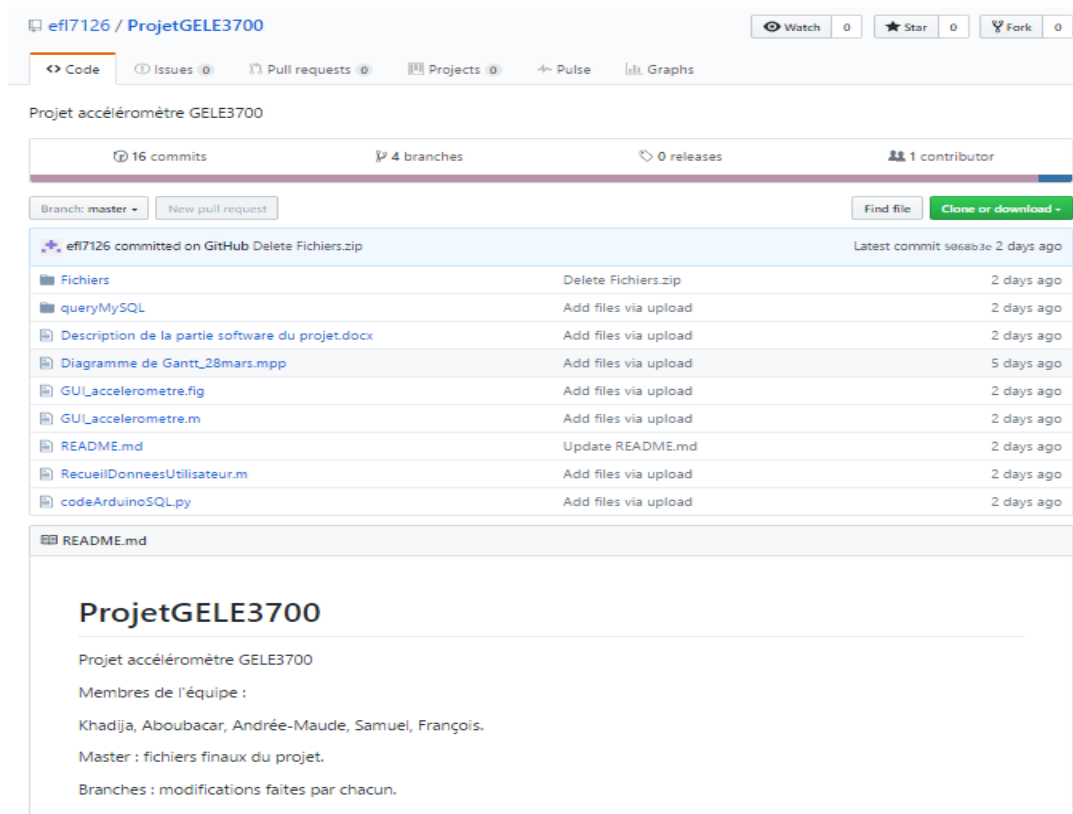


Figure 13 : Aperçu de GitHub lorsque le projet était encore en phase de développement (2 avril 2017)

Sur GitHub, le projet est divisé en six branches :

- Arduino : codes C pour le microcontrôleur.
- Python : codes écrits en Python (notamment pour tester le recueil de données et l'envoi sur le port série ou tester la réception UDP).
- GUI : tous les fichiers écrits pour l'interface graphique sur Matlab.
- MatlabUDP : codes écrits pour tester la réception UDP sur Matlab.
- Processing : tous les codes écrits en Java avec Processing.
- Master : rassemblement de tous les codes des différentes branches.

Une branche correspond à une version provisoire de code. Dans ce projet, chaque branche est reliée à un langage de programmation ou à une section spécifique du projet. Chaque membre de l'équipe est responsable d'inclure les codes à l'intérieur des bonnes branches.

Par la suite, lorsqu'un code est finalisé, les membres de l'équipe peuvent se rassembler et décider si le code peut être envoyé vers la branche Master (avec un *commit*). Après avoir effectué un *commit*, le code est inclus dans la version finale du projet.

4.3 Choix de la base de données

Il a été décidé d'inclure une base de données dans la partie logicielle du projet puisque le montant de données recueillies par les capteurs devenait très important. Il était également important de classer les données afin d'y accéder de manière efficace par la suite. Par exemple, il est facile, avec une base de données, d'accéder aux données relatives à un coureur ou correspondant à un certain intervalle de temps. Si plusieurs appareils sont utilisés en même temps, il est également possible de classer les données pour chacun des appareils.

La base de données choisie pour le projet est **mySQL**. Il s'agit d'un Système de gestion de bases de données relationnelles (SGBDR) qui utilise le langage SQL. mySQL est l'un des systèmes de base de données les plus populaires. Il a l'avantage d'être gratuit et d'être bien adapté pour recueillir des données provenant de capteurs. Des fonctions de requête sont disponibles dans plusieurs langages de programmation, tels Processing et Matlab qui sont utilisés dans ce projet.

mySQL permet essentiellement de manipuler des données, c'est-à-dire sélectionner et afficher des informations enregistrées dans une table, modifier des données, en ajouter ou en supprimer. Il est basé sur le modèle client – serveur. Ce modèle implique que la base de données se trouve sur un serveur qui ne sert qu'à cela, et pour interagir avec cette base de données, il faut utiliser un logiciel "client" qui va interroger le serveur et transmettre la réponse que le serveur lui aura donnée (source : OpenClassrooms.com). Dans ce projet, les logiciels serveur et client seront installés sur un même ordinateur qui servira à recueillir et à analyser les données. C'est donc le mode "localhost" qui sera utilisé.

4.4 Présentation des codes pour essais préliminaires

Tout au long du projet, des codes préliminaires ont été écrits afin de tester l'architecture proposée à la Figure 12. Comme dans tout projet d'ingénierie, certains éléments ont dû être modifiés afin de résoudre des problèmes ou améliorer les performances.

4.4.1 Recueil des données du microcontrôleur sur le port série

Au début du projet, lorsque la base de données mySQL a été mise en place et que la GUI Matlab était en processus de conception, il était très utile d'obtenir des échantillons de données pour vérifier les codes. Vu que le capteur MPU-6050 et le microcontrôleur ESP Thing n'étaient pas encore arrivés, il fallait obtenir des données d'une autre façon. La solution a été d'utiliser un simple capteur de température LM35 qui envoie la température ambiante sur un port analogique d'un Arduino UNO. Étant

donné que la communication sans fil n'était pas encore implémentée, les données étaient simplement envoyées sur le port série de l'ordinateur (Figure 14).

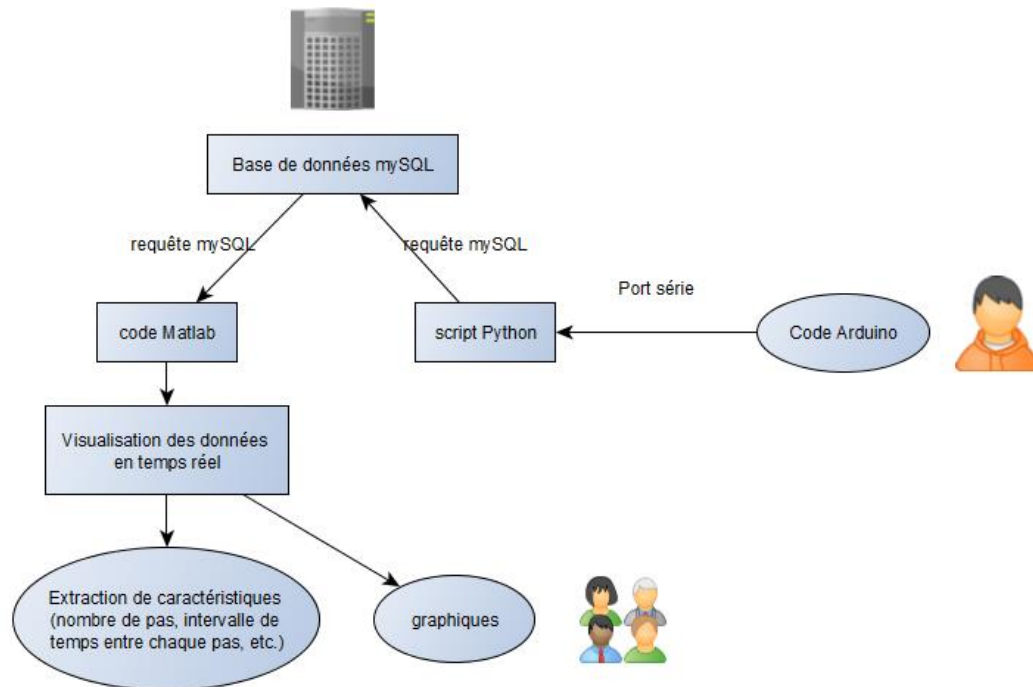


Figure 14 : Envoi de données de température (capteur LM35) sur un port série et enregistrement dans la base de données

Comparé à l'architecture de la Figure 13, les seules différences sont :

- Le code Arduino qui reçoit les données du capteur LM35 (Figure 15)
- La communication par port série
- Le code Python envoyant les données vers mySQL

```

Essai_LM35
float tempC;
int reading;
int tempPin = 0;

void setup()
{
    analogReference(INTERNAL);
    Serial.begin(9600);
}

void loop()
{
    reading = analogRead(tempPin);
    tempC = reading / 9.31;
    Serial.println(tempC);
    delay(1000);
}

```

Figure 15 : Code Arduino qui recueille les données de température du capteur LM35

Un script a été écrit en Python pour recueillir les données et les envoyer vers MySQL. Ce script demande tout d'abord à l'utilisateur d'entrer un numéro d'utilisateur, qui servira à classer les données dans la base de données. Les résultats d'exécution du script sont présentés à la Figure 16.

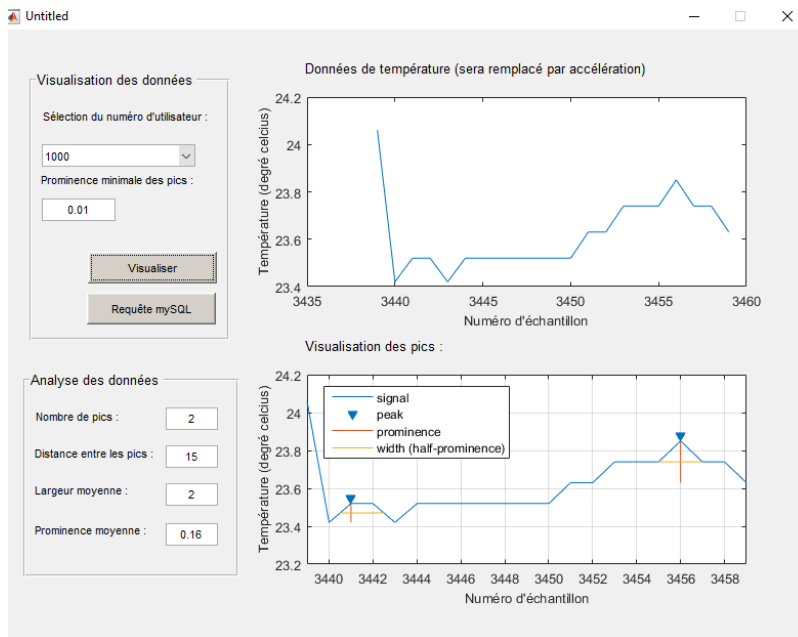
```

C:\Users\Francois\Documents\Hiver 2017\GELE3700 - Projet GE 1\Fichiers\Python>py
thon codeArduinoSQL.py
Entrez un numéro d'utilisateur : 32
Trying... COM3
Successfully connected on port : COM3
23.20
Envoi réussi a MySQL
22.66
Envoi réussi a MySQL
22.77
Envoi réussi a MySQL
22.66
Envoi réussi a MySQL
22.56
Envoi réussi a MySQL
22.66
Envoi réussi a MySQL

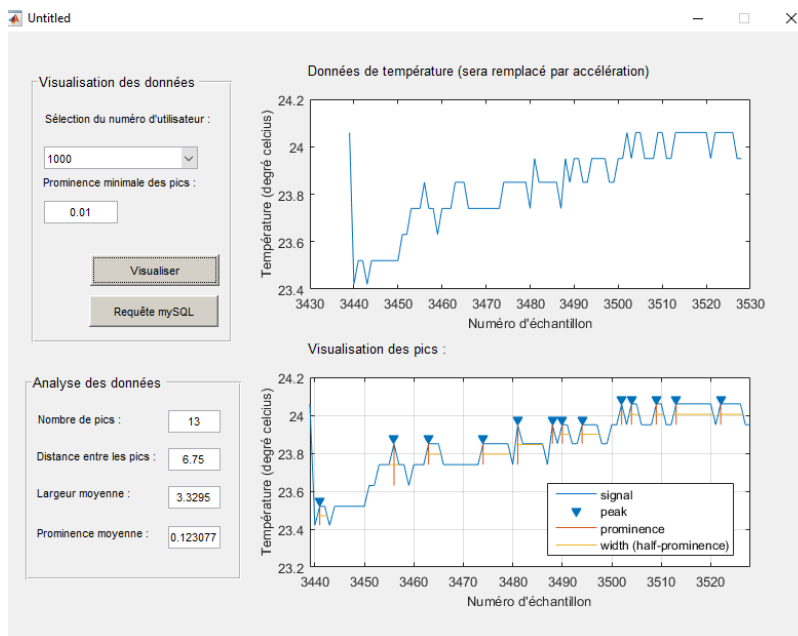
```

Figure 16 : Script Python pour le recueil de données sur le port série

Finalement, une version provisoire de l'interface graphique a été codée sur Matlab pour recueillir les données enregistrées dans la base de données. La version finale de ce code Matlab est présentée plus tard dans ce rapport. Les résultats obtenus sont présentés à la Figure 17.



Temps t0



Temps t1

Figure 17 : Interface graphique provisoire sur Matlab. L'image du haut montre les données de température recueillies à un temps t0. L'image du bas montre les données de température recueillies à un temps t1, après avoir appuyé sur le bouton "requête mySQL" et ensuite sur le bouton "visualiser".

Ces tests montrent que les codes provisoires fonctionnent correctement. Aucune erreur n'est obtenue et les résultats sont satisfaisants. Ces codes provisoires seront ensuite utilisés pour tester l'envoi de données sur Internet avec le protocole UDP.

4.4.2 Réception UDP sur Matlab

L'objectif initial du projet était d'abandonner le code Python (qui permet la réception sur le port série) et de le remplacer par une fonction, à l'intérieur de la GUI Matlab, qui agirait comme client UDP. Ceci permettrait aux chercheurs du laboratoire de G. Handrigan de seulement avoir à ouvrir la GUI Matlab. Ils n'auraient donc pas besoin d'exécuter un programme séparé pour le recueil des données. L'architecture proposée est présentée à la Figure 18.

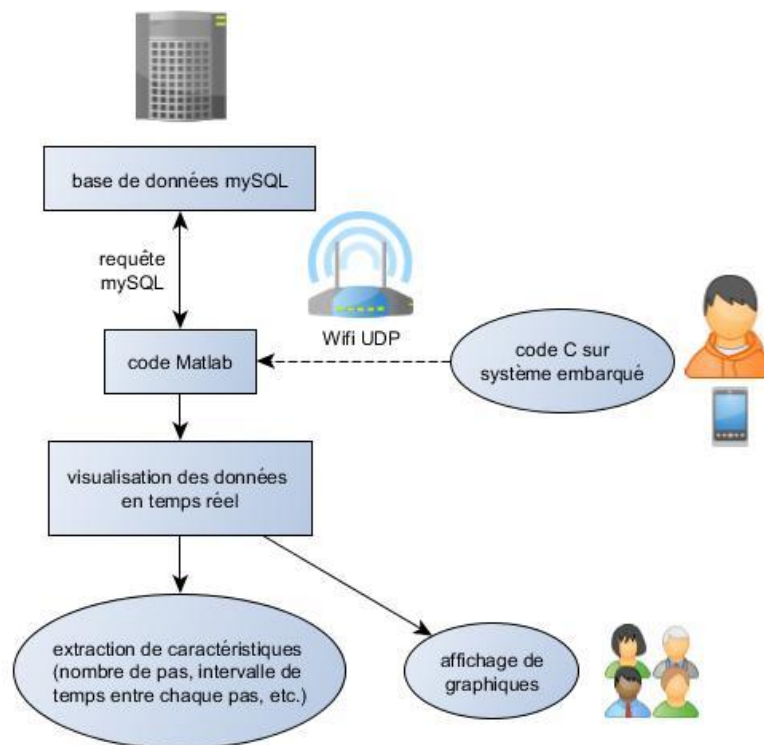


Figure 18 : Client UDP implémenté sur Matlab

La classe `udp`, implémentée dans le toolbox Matlab Instrument Control Toolbox, a été utilisée pour créer un client UDP. L'extraction des données a été codée dans le fichier `Pilote_ConnectionUDP.m`, qui est disponible sur GitHub (voir annexe, section 4.1.3).

La fonction a été testée avec les données d'accélération envoyées par l'émetteur WiFi du microcontrôleur ESP Thing. Malheureusement, l'équipe a toujours observé d'importants problèmes de délai à la réception. Par exemple, un délai maximal fixé à 0,5 seconde était fréquemment dépassé lors des essais, ce qui est inacceptable pour rencontrer les exigences du laboratoire de recherche. La Figure 19 montre le type d'erreur qui était rencontrée.

```
Warning: Unsuccessful read: The specified amount of data was not returned within the Timeout period.  
UdpData =
```

Figure 19 : Erreur de délai lors de la réception UDP sur Matlab

Plusieurs stratégies ont été envisagées afin de régler cette erreur. Tout d'abord, lorsque la fonction a initialement été testée, chaque paquet envoyé contenait 1 octet. Ceci était possiblement inefficace, comparé à si on envoyait plus d'octets par paquet. Le code a donc été modifié afin que chaque paquet envoyé contienne plutôt 64 octets. Malheureusement, le problème était toujours observé à plusieurs reprises. La longueur des paquets a également été testée à 128 octets, sans améliorer significativement les résultats.

Face à ces problèmes, l'équipe a pensé que la connexion au réseau umcm-projets était peut-être de mauvaise qualité. Il a été décidé de laisser tomber Matlab pour la réception des données UDP. Le code Python écrit à la section 4.4.1 a donc été réutilisé. Une fonction de réception sur port UDP a été ajoutée à ce code, qui a ensuite été testé. Malheureusement, l'équipe n'est jamais arrivée à obtenir de connexion sur le réseau umcm-projet.

En dernier recours, une autre implémentation du client UDP a été testée avec les bibliothèques Processing pour Java. Cette implémentation a finalement permis à l'équipe de recevoir les données sur UDP sans problème de délai apparent. Les détails concernant ce code sont présentés à la section 4.5.1.

4.5 Présentation du concept final

4.5.1 Création et utilisation d'une base de données sur MySQL

4.5.1.1 Installation

La base de données utilisée dans ce projet est le système MySQL. La première étape a été de télécharger MySQL à partir du site Web de l'entreprise (<https://dev.mysql.com/downloads/mysql/#downloads>). Le serveur et le client MySQL ont été installés sur le même ordinateur (localhost). Lors de l'installation, un mot de passe a été défini pour l'utilisateur par défaut de MySQL, l'utilisateur "root" (il est déconseillé de ne pas définir de mot de passe).

La deuxième étape de l'installation est de définir un chemin vers MySQL aux dossiers explorés par l'invité de commande de Windows. Pour ce faire, la commande suivante doit être exécutée dans l'invité de commande, en remplaçant chemin_vers_mysql_bin par la localisation de mysql\bin sur l'ordinateur :

```
>>> set PATH=%PATH%;chemin_vers_mysql_bin
```

Sur l'ordinateur utilisé, la commande suivante a été exécutée dans l'invité de commande :

```
>>> set PATH=%PATH%;C:\Program Files\MySQL\MySQL Server 5.7\bin
```

4.5.1.2 Connexion et déconnexion du client

La commande nécessaire pour se connecter au client est la suivante :

```
>>> mysql -h localhost -u root -pmotdepasse
```

La commande mysql démarre le client. Il faut lui donner 3 paramètres :

- La localisation du serveur (localhost). Étant donné que le serveur et le client sont installés sur le même ordinateur, il n'est pas nécessaire de le spécifier, puisque localhost est la valeur par défaut.
- Le nom de l'utilisateur (root).
- Le mot de passe déterminé à l'installation.

Pour déconnecter le client, il faut utiliser la commande suivante :

```
>>> quit
```

Finalement, il est également important de définir l'encodage utilisé, UTF-8. La commande suivante est donc requise à chaque connexion :

```
>>> SET NAMES 'utf8';
```

Au lieu de toujours avoir à taper cette commande, on peut définir l'encodage lors de la connexion au client :

```
>>> mysql -u root -p --default-character-set=utf8
```

4.5.1.3 Création d'une base de données

La commande permettant de créer la base de donnée est la suivante :

```
>>> CREATE DATABASE acceleration CHARACTER SET 'utf8';
```

Cette commande crée une base de données appelée acceleration et définit l'encodage utilisé (utf8).

Pour utiliser la base de donnée, il faut entrer la commande suivante :

```
>>> USE acceleration;
```

4.5.1.4 Création d'une table

Sur MySQL, les bases de données sont organisées en tables. Une table est une organisation de données comportant plusieurs colonnes. Dans notre cas, chaque colonne représente une différente mesure et chaque rangée correspond à un temps précis.

Sur mySQL, Les colonnes sont associées à un type de données et à une autorisation de contenir aucune valeur (NULL).

La table utilisée pour ce projet doit permettre de recueillir les données d'accélération (sur l'axe X, Y et Z) et les données du gyroscope (sur l'axe X, Y, Z). Il y a donc, au minimum, 6 colonnes à insérer dans la table. Les autres colonnes qu'il est avantageux d'ajouter sont les suivantes :

- **id.** Il s'agit d'un identifiant pour chaque donnée de la table. id est incrémenté de 1 à chaque ajout. Par conséquent, une donnée avec un id de 29 a été mesurée avant une donnée avec un id de 30. De plus, si des données sont ajoutées après une première série de mesures, l'id commence à partir du précédent.
- **utilisateur.** Cette colonne correspond à un numéro d'utilisateur déterminé par la personne qui utilise le programme Processing. Le numéro d'utilisateur pourrait correspondre aux différents coureurs qui vont participer dans l'étude.
- **temps.** Cette colonne correspond à l'horodatage (timestamp) associée à chaque mesure, défini lors de la réception dans le programme Processing.

La table utilisée dans ce projet a été créée en exécutant la commande suivante :

```
>>> CREATE TABLE acceleration4 (  
    id INT(11) NOT NULL AUTO_INCREMENT,  
    utilisateur INT(11),  
    temps datetime,  
    accX DOUBLE,  
    accY DOUBLE,  
    accZ DOUBLE,  
    gyroX DOUBLE,  
    gyroY DOUBLE,  
    gyroZ DOUBLE,  
    PRIMARY KEY (id)  
);
```

Quelques précisions doivent être faites. Tout d'abord, le type utilisé pour les colonnes id et utilisateur sont INT(11). Le nombre entre parenthèses spécifie la taille d'affichage du type, mais n'influence pas son encodage (un int restera toujours 4 octets). Pour la colonne temps, le type utilisé est datetime; il s'agit d'un type spécialisé de mySQL qui permet de stocker une année, un mois, un jour, une heure, une minute et une seconde selon le format suivant :

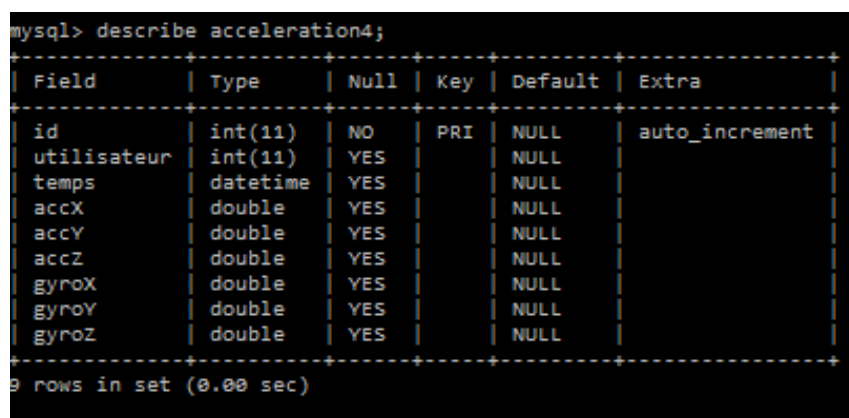
```
'AAAA-MM-JJ HH:MM:SS'
```

Les colonnes stockant les données d'accélération et de gyroscope sont tous définies avec le type double. La spécification NOT NULL AUTO_INCREMENT pour la colonne id indique que l'id sera auto-incrémenté à chaque ajout dans la table, et que la colonne ne peut pas inclure une donnée nulle. La spécification PRIMARY KEY (id), à la fin de la commande, indique que la colonne id sert d'identifiant pour chaque donnée et qu'elle doit contenir des valeurs uniques.

Pour visualiser le format de la table, la commande suivante doit être exécutée :

```
>>> describe acceleration4
```

Le résultat obtenu est présenté à la Figure 20.



```
mysql> describe acceleration4;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
utilisateur	int(11)	YES		NULL	
temps	datetime	YES		NULL	
accX	double	YES		NULL	
accY	double	YES		NULL	
accZ	double	YES		NULL	
gyroX	double	YES		NULL	
gyroY	double	YES		NULL	
gyroZ	double	YES		NULL	

9 rows in set (0.00 sec)

Figure 20 : Format de la table utilisée pour stocker les données

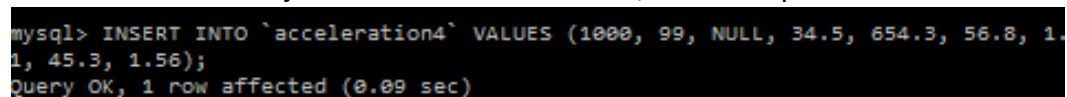
Cette table sera utilisée pour stocker les données reçues du microcontrôleur par le code Processing.

4.5.1.5 Ajout de données et consultation de la table

L'ajout de données dans la table peut être fait à partir de l'invité de commande ou en utilisant une API provenant d'une bibliothèque. La syntaxe reste la même dans les deux cas. La commande à exécuter est la suivante :

```
>>> INSERT INTO `acceleration4` VALUES (1000, 99, NULL, 34.5, 654.3, 56.8, 1.1, 45.3, 1.56);
```

Les valeurs spécifiées comme arguments entre parenthèses ne sont que des exemples de valeurs. Si l'ajout a bien fonctionné, une réponse semblable à la



```
mysql> INSERT INTO `acceleration4` VALUES (1000, 99, NULL, 34.5, 654.3, 56.8, 1.1, 45.3, 1.56);
Query OK, 1 row affected (0.09 sec)
```

Figure 21 devrait être obtenue.

```
mysql> INSERT INTO `acceleration4` VALUES (1000, 99, NULL, 34.5, 654.3, 56.8, 1.1, 45.3, 1.56);
Query OK, 1 row affected (0.09 sec)
```

Figure 21 : Ajout d'une donnée dans la table et réponse

Pour afficher les données stockées à l'intérieur de la table, il suffit d'exécuter la commande suivante :

```
>>> SELECT * FROM acceleration4;
```

La Figure 22 montre l'affichage de la table avec la donnée tout juste insérée.

```
mysql> SELECT * FROM acceleration4
-> ;
+-----+-----+-----+-----+-----+-----+-----+-----+
| id   | utilisateur | temps | accX | accY | accZ | gyroX | gyroY | gyroZ |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1000 |          99 | NULL  | 34.5 | 654.3 | 56.8 | 1.1   | 45.3  | 1.56  |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Figure 22 : Affichage de la table

4.5.2 Code Processing pour le recueil de données et l'envoi vers MySQL

Une fois la base de donnée créée, il fallait établir un lien entre l'envoi de données sur UDP par le microcontrôleur et le stockage dans la base de données. Comme présenté à la section 4.4.2, la conception initiale comprenait une réception au niveau de Matlab. Malheureusement, quand cela a été essayé, il y a avait souvent des temps d'attente trop longs pour remplir les objectifs du projet. La solution de rechange a été d'utiliser un programme séparé codé sur Processing. Processing est un langage de programmation basé sur Java qui comprend des bibliothèques supplémentaires, incluant une bibliothèque pour la communication UDP. Un ordigramme du code conçu est présenté à la Figure 23 et est discuté dans les section suivantes.

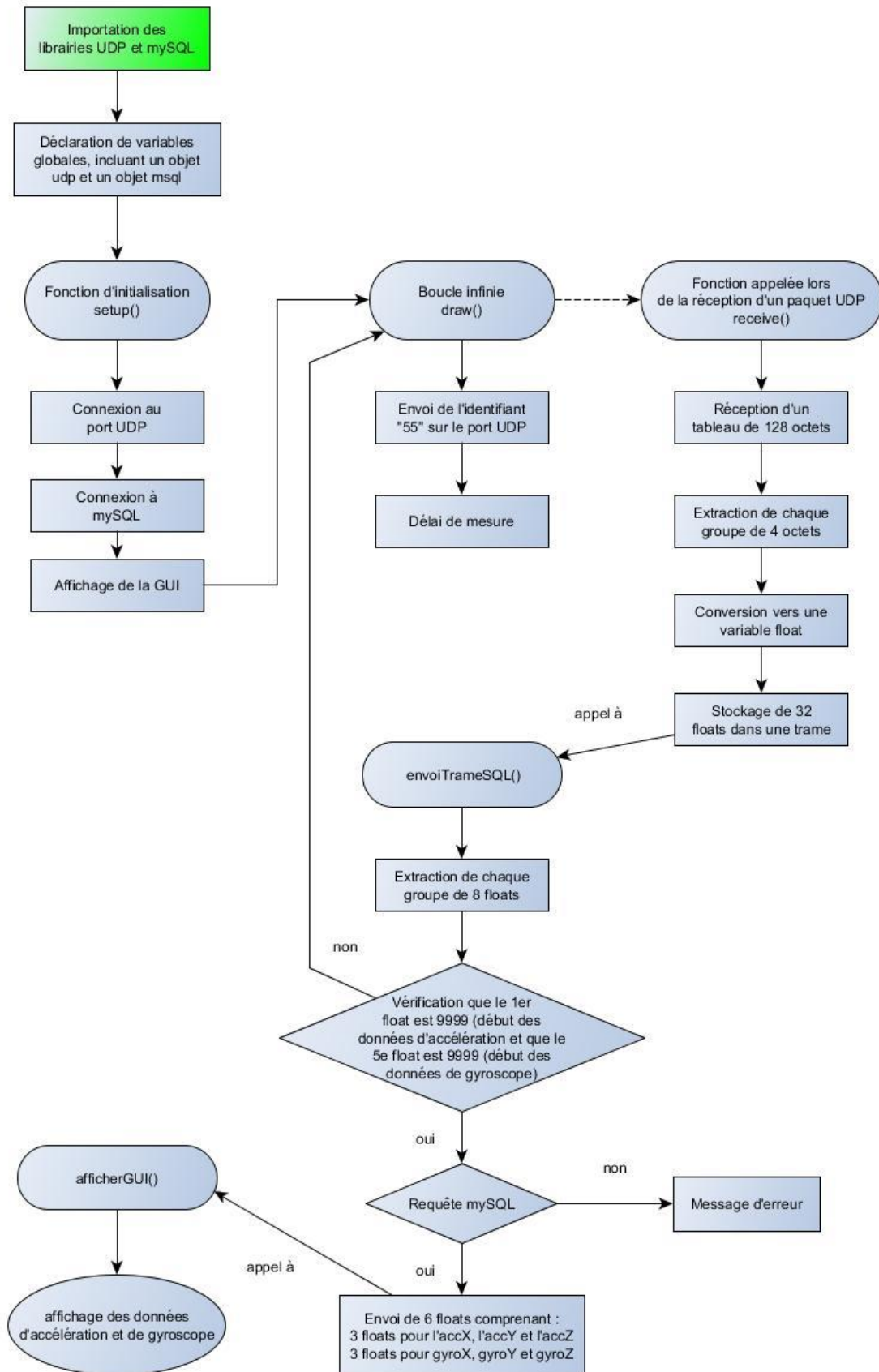


Figure 23 : Ordinogramme du code Processing pour la réception UDP et l'envoi vers mySQL

4.5.2.1 Réception par protocole UDP

L'une des tâches principales du code est de recevoir les données envoyées par le microcontrôleur sur le port UDP. Pour ce faire, un objet udp est instancié à partir de la librairie Processing UDP hypermedia.net. La connexion au port UDP est ensuite effectuée avec les informations du réseau umcm-projet :

```
Ip = "10.5.64.33"  
Port = 2640
```

Une fois ces configurations complétées, le programme entre dans la boucle infinie de la fonction draw(). Pour que le microcontrôleur envoie une trame de donnée, il faut que le code Processing envoie l'identifiant "55" sur le port UDP. Après cet envoi, un délai est appliqué pour permettre la réception. Ce délai détermine la fréquence de recueil des données.

Lorsque des données sont reçues sur le port UDP (envoyées par le microcontrôleur), la fonction receive() est automatiquement appelée par Processing. Cette fonction recueille les données dans un buffer. Au total, un paquet de 128 octets est envoyé à chaque envoi par le code du microcontrôleur. Étant donné que les valeurs mesurées par le MPU-6050 sont encodées en type float, ces 128 octets représentent réellement 32 floats (4 octets par float).

L'ordre des octets dans chaque trame des paquets est présenté au Tableau 14. Les rangées en bleu représentent le numéro de l'octet; on voit que chaque float comprend bien 4 octets. Chaque trame de donnée est composée de 8 floats. Ceci implique que chaque paquet envoyé contient 4 trames (4 trames x 8 floats x 4 octets = 128 octets).

La trame est encodée comme ceci : elle comprend d'abord un identifiant (9999) pour commencer avec les données d'accélération. Il y a alors les accélérations sur X, Y et Z. Un deuxième identifiant permet de commencer les données du gyroscope. Il y a alors les données du gyro sur X, Y et Z.

Tableau 14 : Encodage de la trame envoyée sur UDP

B0	B1	B2	B3	B4	B5	B6	B7
Float 1	Float 1	Float 1	Float 1	Float 2	Float 2	Float 2	Float 2
Identifiant 1 (9999)	->	->	->	AccX (125)	->	->	->
B8	B9	B10	B11	B12	B13	B14	B15
Float 3	Float 3	Float 3	Float 3	Float 4	Float 4	Float 4	Float 4
AccY (65261)	->	->	->	AccZ (1960)	->	->	->
B16	B17	B18	B19	B20	B21	B22	B23
Float 5	Float 5	Float 5	Float 5	Float 6	Float 6	Float 6	Float 6
Identifiant 2 (9999)	->	->	->	GyroX (103)	->	->	->
B24	B25	B26	B27	B28	B29	B30	B31
Float 7	Float 7	Float 7	Float 7	Float 8	Float 8	Float 8	Float 8
GyroY (65268)	->	->	->	GyroZ (1933)	->	->	->

La fonction `receive()` du code Processing permet d'extraire les floats du Tableau 14 pour ensuite les envoyer vers la base de données `mySQL` et les afficher sur l'interface graphique codée avec Processing. L'affichage des données sur la GUI Processing se fait en temps réel. Un exemple est présenté à la Figure 24.

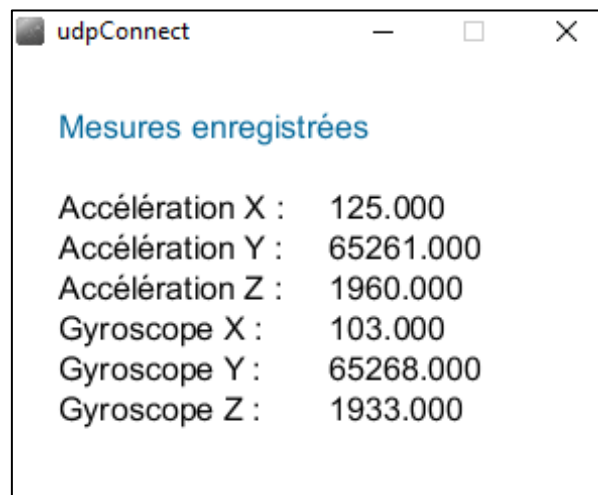


Figure 24 : Exemple d'affichage des données sur la GUI Processing

4.5.2.2 Envoi d'une requête vers MySQL

Lorsque les données sont reçues par protocole UDP et sont extraites sous forme de floats, la fonction `envoiTrameSQL()` est appelée. Cette fonction permet d'envoyer chaque trame de données vers la base de données. Ce sera donc 8 floats, moins les deux identifiants, ce qui donne 6 floats. En effet, la fonction vérifie que le float 1 et 5 de la trame correspondent bien à l'identifiant 9999 (voir Tableau 14). Ceci permet de s'assurer que la trame comprend toutes les données mesurées et qu'aucune donnée n'a été perdue en cours de route. Si cette condition est respectée, les 6 floats de données sont envoyés vers MySQL avec la commande suivante :

```
>>> INSERT INTO `acceleration` VALUES (NULL, %s (no  
d'utilisateur), %s (horodatage), %s (accX), %s (accY), %s  
(accZ), %s (gyroX), %s (gyroY), %s (gyroZ))
```

La Figure 25 montre que la donnée obtenue à la Figure 24 est bien insérée dans la base de données MySQL (encadré bleu).

1976	101	NULL	89	65248	1930	105	65242	1941
1977	101	NULL	105	65228	1951	104	65249	1929
1978	101	NULL	112	65235	1931	109	65256	1924
1979	101	NULL	111	65302	1923	108	65291	1927
1980	101	NULL	109	65307	1918	106	65320	1908
1981	101	NULL	118	65292	1956	113	65272	1950
1982	101	NULL	125	65261	1960	103	65268	1933

Figure 25 : Insertion d'une donnée mesurée dans la base de données MySQL

4.5.3 Design de la GUI sur Matlab

Le développement d'une interface graphique sur Matlab était essentiel pour permettre aux assistants de recherche du laboratoire de Grant Handrigan d'observer et d'analyser les données recueillies. Le design de l'interface ne devait pas être trop complexe, mais devait au moins permettre d'observer les données sous forme graphique lorsqu'elles sont recueillies par un appareil sur un coureur.

Le design de l'interface graphique a été réalisé en utilisant l'utilitaire GUIDE de Matlab. GUIDE est basé sur un principe de *drag and drop*, ce qui facilite l'élaboration d'une interface graphique simple. L'interface conçue avec GUIDE est présentée à la Figure 26.

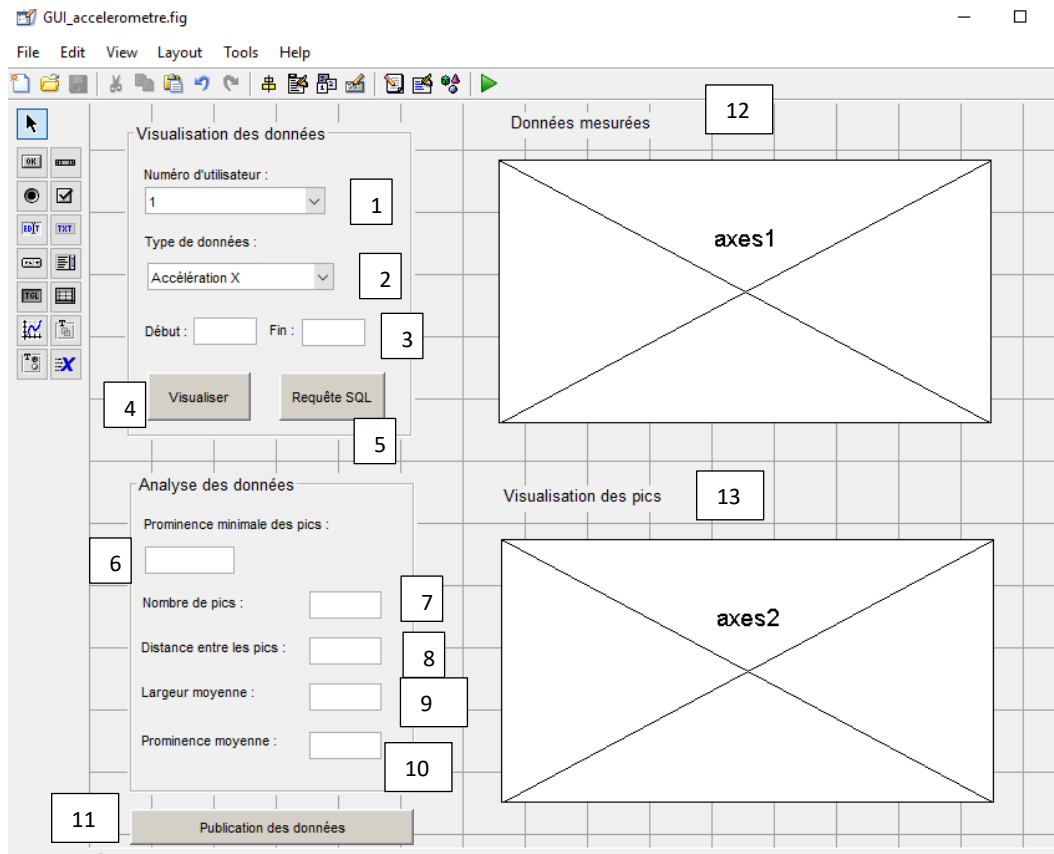


Figure 26 : Conception de l'interface graphique sur GUIDE

Les composantes de l'interface graphique sont présentées ci-dessous :

1. Numéro d'utilisateur : Ce menu déroulant permet de choisir le numéro d'utilisateur tel que défini dans la base de données MySQL. Lorsqu'un numéro est choisi, seules les données correspondant à ce numéro seront affichées. Un numéro d'utilisateur pourrait être associé à chaque coureur de l'expérience.
2. Type de données : Ce menu déroulant donne trois choix d'affichage : l'accélération en X, en Y ou en Z, ou les données du gyroscope en X, en Y ou en Z. Les données correspondant au choix d'affichage seront affichées sur les graphiques lorsque l'utilisateur appuie sur le bouton Visualiser.
3. Début et Fin : Si l'utilisateur désire visualiser les données sur un intervalle restreint de valeurs, il peut spécifier une valeur de début et de fin dans ces boîtes. Lorsqu'il appuie ensuite sur le bouton Visualiser, les données seront seulement affichées pour l'intervalle spécifié. Si l'utilisateur désire revenir à l'intervalle complet, il n'a qu'à laisser ces cases vides.
4. Bouton visualiser : Affiche les données dans le graphique 12 et l'identification des pics dans le graphique 13.
5. Bouton requête SQL : Lorsque l'utilisateur appuie sur ce bouton, Matlab effectue une requête sur la base de données MySQL. Les nouvelles données sont donc mises à jour. Ceci permet aux assistants de laboratoire d'obtenir les

données des coureurs dans un mode presque à temps réel. Il faut ensuite peser sur le bouton visualiser pour observer les données mises à jour.

6. Prominence minimale des pics : la prominence d'un pic est une mesure de la hauteur d'un pic en relation avec les pics à proximité. L'utilisateur peut déterminer une prominence minimale en spécifiant une valeur dans la case 6.
7. Nombre de pics : cette case affiche le nombre de pics identifiés.
8. Distance moyenne entre les pics : cette case affiche la distance moyenne entre chaque pic identifié.
9. Largeur moyenne des pics : cette case affiche la largeur moyenne, à mi-hauteur, de tous les pics identifiés.
10. Prominence moyenne : cette case affiche la prominence moyenne des pics.
11. Publication des données : ce bouton, lorsqu'appuyé par l'utilisateur, publie les données dans un fichier .csv (valeurs séparées par une virgule). Les données publiées sont filtrées en fonction du numéro d'utilisateur, du type de données et de l'intervalle de données choisi. Cette fonction est importante pour le laboratoire de recherche de Grant Handrigan puisqu'elle permet de filtrer leurs données. En effet, pour chaque coureur, un nombre important de données sera recueilli. Par la suite, les assistants peuvent utiliser leur propre logiciel pour l'analyse des données.
12. Graphique de données mesurées : affichage des données mesurées.
13. Visualisation des pics : affichage des pics identifiés.

La section Analyse des données de la GUI comprend plusieurs métriques qui pourraient être utiles pour le projet de recherche de G. Handrigan. En effet, l'identification des pics est très semblable à celle des pas effectués par les coureurs. Le nombre de pics devrait correspondre au nombre de pas, tout comme la distance moyenne entre les pics, leur largeur ou leur prominence moyenne. Selon les données de l'étude, il sera important d'ajuster la valeur de prominence minimale d'identification des pics. Si celle-ci est trop haute, certains pas ne seront pas identifiés, alors que si elle est trop basse, le bruit sera augmenté.

Finalement, le bouton Publication des données permet au client d'effectuer ses propres analyses à partir de données en format .csv. Il pourrait, par exemple, utiliser un logiciel comme Excel pour lire ces données.

4.5.4 Présentation du code Matlab

Le code de la GUI Matlab est présenté dans l'ordinogramme de la Figure 27.

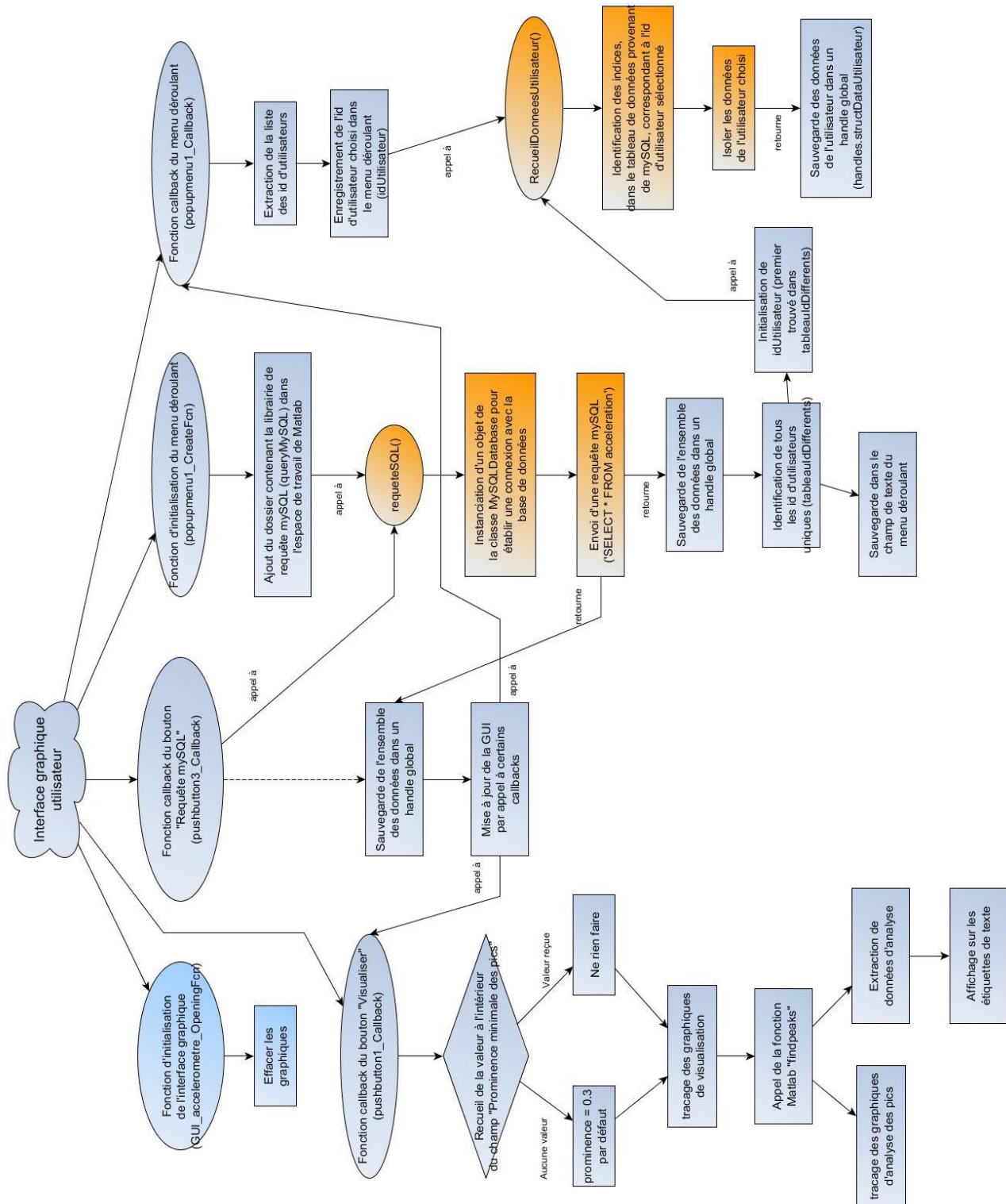


Figure 27 : Ordinogramme du code de la GUI Matlab

Lorsque l'interface graphique est initialisée une première fois, des fonctions spécialisées de Matlab, appelées *CreateFcn*, sont appelées. Pour répondre aux besoins de l'interface graphique, du code a été ajouté dans plusieurs de ces fonctions. Un deuxième type de fonction spécialisée sont les *callbacks*. Ces fonctions sont appelées à chaque fois que l'utilisateur appuie sur l'élément correspondant de l'interface graphique.

Le code ne sera pas décrit en détails puisque beaucoup du code est seulement utilisé pour l'affichage graphique. La fonction `requeteSQL()` mérite toutefois un peu d'explications.

4.5.4.1 Fonction `requeteSQL()`

La fonction `requeteSQL()` fait appel aux classes `MySQLDatabase` fournies par Matlab. Lorsque la GUI est ouverte une première fois, la totalité des données de la table `acceleration` est importée, à partir de la base de données. Ceci pourrait prendre un peu de temps si la table contient beaucoup de données. Cela n'a pas été vu comme un point négatif puisque ça se produit seulement à l'ouverture de la GUI. La commande `mysql` utilisée est la suivante :

```
>>> SELECT * FROM acceleration;
```

Lorsque la GUI est ouverte et qu'elle est utilisée pour voir les données en temps réel, il devient important de limiter le temps d'attente. Par conséquent, à chaque fois que l'utilisateur change la sélection dans le menu déroulant du numéro d'utilisateur, ou qu'il clique sur le bouton requête SQL, une requête spécifique est effectuée :

```
>>> SELECT * FROM acceleration WHERE utilisateur = "id
utilisateur";
```

Cette commande permet d'obtenir les données associées à un utilisateur en particulier. En effet, lors des expériences, les données proviendront d'un seul utilisateur en même temps. Il est donc inutile d'importer la base de données au complet à chaque fois.

Chapitre 5 : Résultats

5.1 Boîtier et circuit imprimé

Dû à des contraintes hors du contrôle de l'équipe, le boîtier réalisé sur SolidWorks et le circuit imprimé réalisé sur Eagle n'ont pas pu être produits par le technicien. L'appareil complété n'a donc pas pu être testé. L'appareil sera donc présenté lors de la présentation finale du projet.

Pour l'instant, un prototype sur planchette a été réalisé afin de tester le fonctionnement des codes (Figure 28).

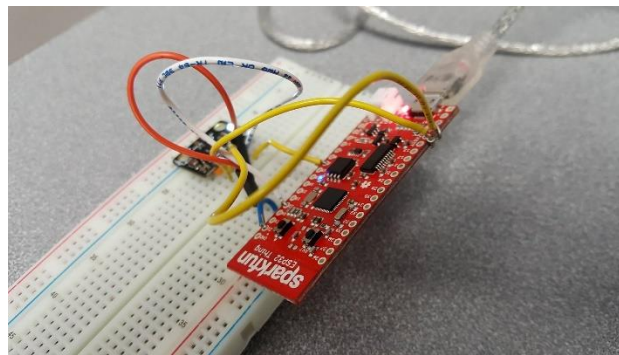


Figure 28 : Prototype de l'appareil utilisé pour tester les codes

5.2 Logiciel sur système embarqué et envoi UDP

La Figure 29 montre un exemple de données reçues sur le port série du système embarqué. Ces données indiquent que le microcontrôleur réussit bien à se connecter au réseau umcm-projet par protocole UDP. De plus, les données sont correctement reçues à partir capteur MPU-6050. L'envoi UDP est confirmé par la réception des paquets à la section 5.3 Réception UDP avec Processing et envoi vers la base de données.

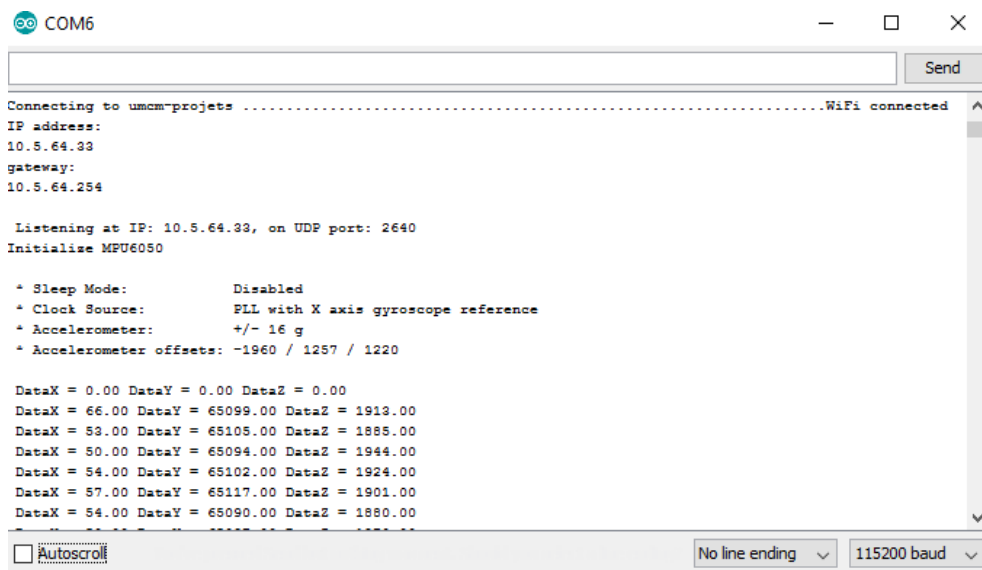


Figure 29 : Données envoyées par le système embarqué sur le port série

5.3 Réception UDP avec Processing et envoi vers la base de données

La réception UDP avec Processing fonctionne correctement, tel que montré à la Figure 24 : Exemple d’affichage des données sur la GUI Processing. Le problème de temps d’attente pour la transmission UDP semble avoir été réglé en utilisant Processing.

L’envoi et la lecture dans la base de données mySQL fonctionnent également, tel que montré à la Figure 25 : Insertion d’une donnée mesurée dans la base de données mySQL.

Un exemple de fonctionnement, qui combine les Figure 24 et Figure 25, est présenté ci-dessous (Figure 30).

udpConnect									
Mesures enregistrées									
Accélération X :	125.000								
Accélération Y :	65261.000								
Accélération Z :	1960.000								
Gyroscope X :	103.000								
Gyroscope Y :	65268.000								
Gyroscope Z :	1933.000								

1976	101	NULL	89	65248	1930	105	65242	1941
1977	101	NULL	105	65228	1951	104	65249	1929
1978	101	NULL	112	65235	1931	109	65256	1924
1979	101	NULL	111	65302	1923	108	65291	1927
1980	101	NULL	109	65307	1918	106	65320	1908
1981	101	NULL	118	65292	1956	113	65272	1950
1982	101	NULL	125	65261	1960	103	65268	1933

Figure 30 : Exemple de fonctionnement de la réception UDP et de l’envoi vers mySQL

5.4 Interface graphique

Le test de l'interface graphique est concluant. La requête MySQL, l'extraction des données pour chaque utilisateur et l'affichage d'un type de données fonctionnent correctement. La Figure 31 montre un exemple de résultat obtenu avec le numéro d'utilisateur 101, le type "accélération Z" et un choix de numéro d'échantillon entre 1600 et 1900. Il ne s'agit pas des données de course, mais plutôt de données acquises avec le prototype sur une table. On remarque que 2 pics importants sont obtenus (ce qui pourrait correspondre à 2 pas).

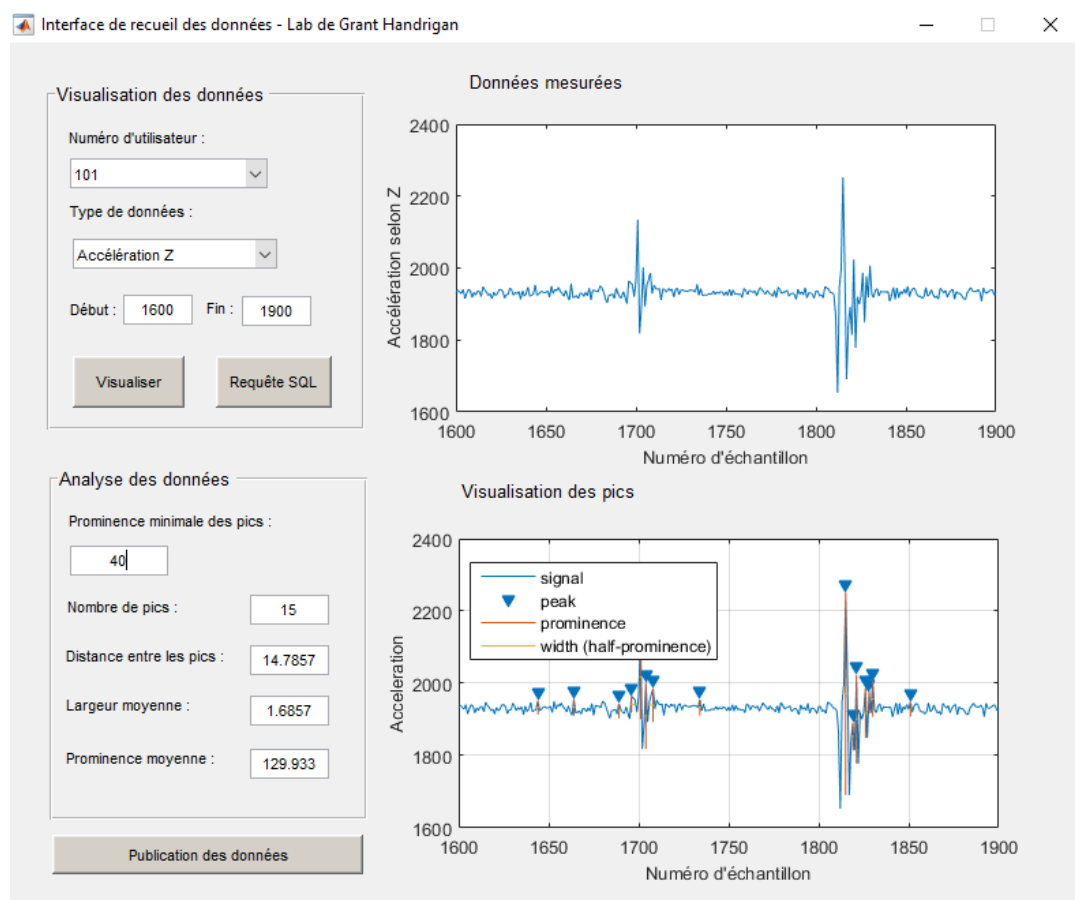


Figure 31 : Test de la GUI avec une prominence minimale des pics de 40

Avec une prominence minimale de 40, le nombre de pics identifiés est 15 pics, ce qui est trop élevé (certains des pics identifiés correspondent à du bruit).

Si on augmente la prominence minimale à 60, le nombre de pics diminue à 8 (Figure 32).

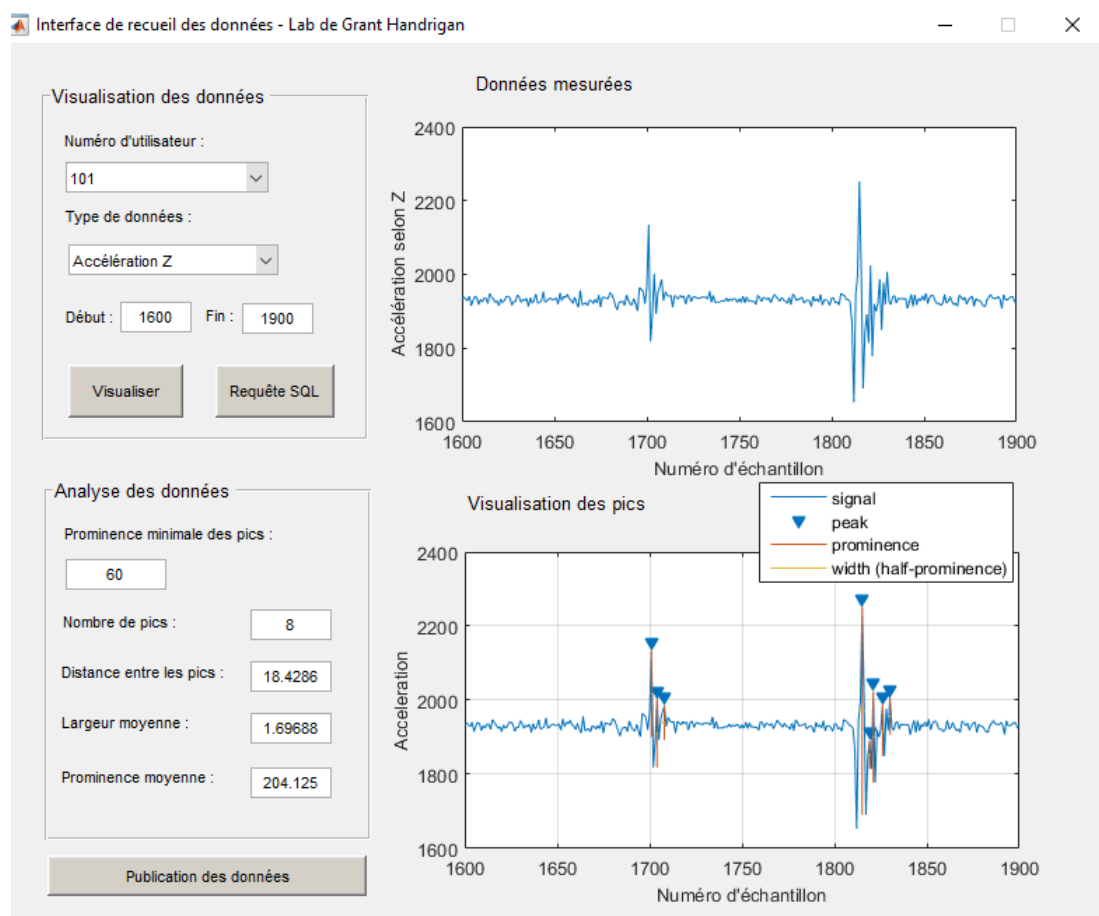


Figure 32 : Diminution du nombre de pics avec une augmentation de la prominence minimale

Pour obtenir une identification optimale des pas, il faut augmenter la prominence minimale jusqu'à 200. Le système identifie alors les deux pics principaux en plus d'un autre pic (bruit).

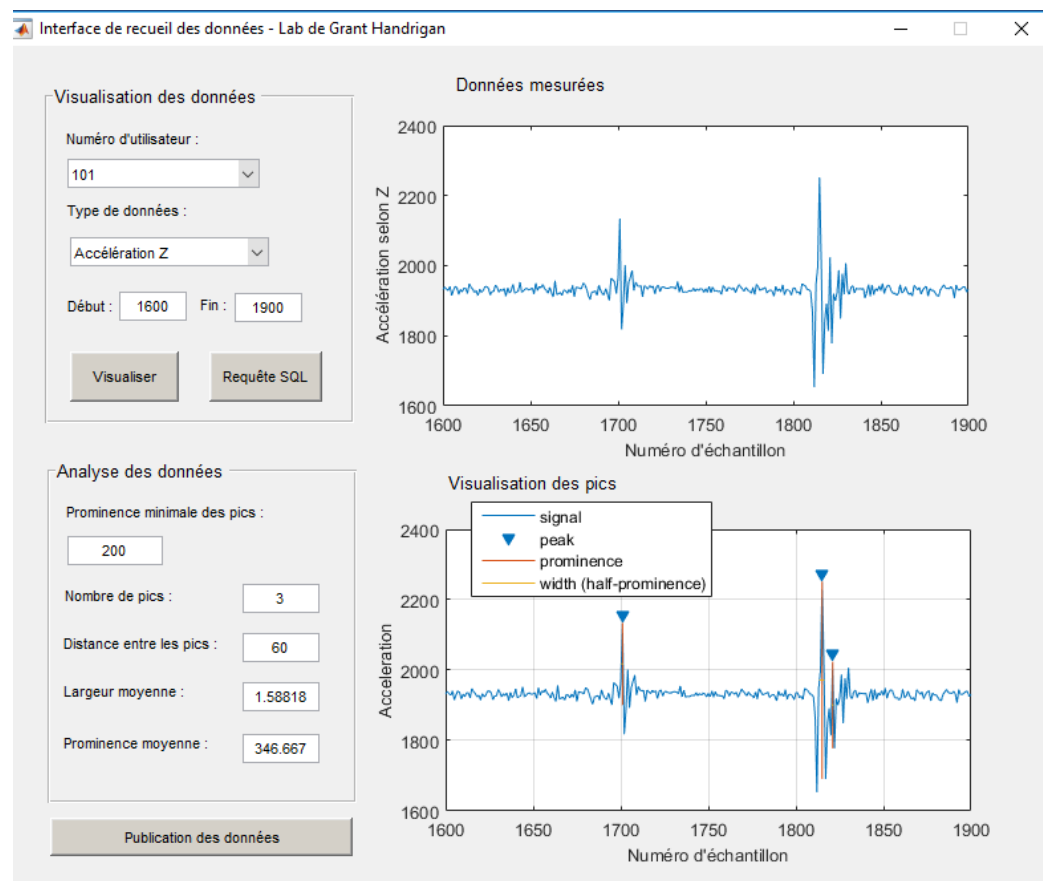


Figure 33 : Identification des pas en utilisant une prominence minimale de 200

Les résultats montrent donc que le système conçu par l'équipe fonctionne correctement et qu'il permettrait bien d'identifier les pas d'un coureur lors d'expériences du laboratoire de G. Handrigan.

5.5 Alimentation et batterie

Les tests pour l'affichage du niveau de la batterie a pu être effectuée avec le prototype. La figure 36 montre un bon fonctionnement de la LED et la figure 35 montre la tension d'entrée qui représente la tension lue au borne de la batterie. on voit bien qu'une lumière bleu s'affiche du au pladge de valeur definie precedemmant pour la plage de tension.

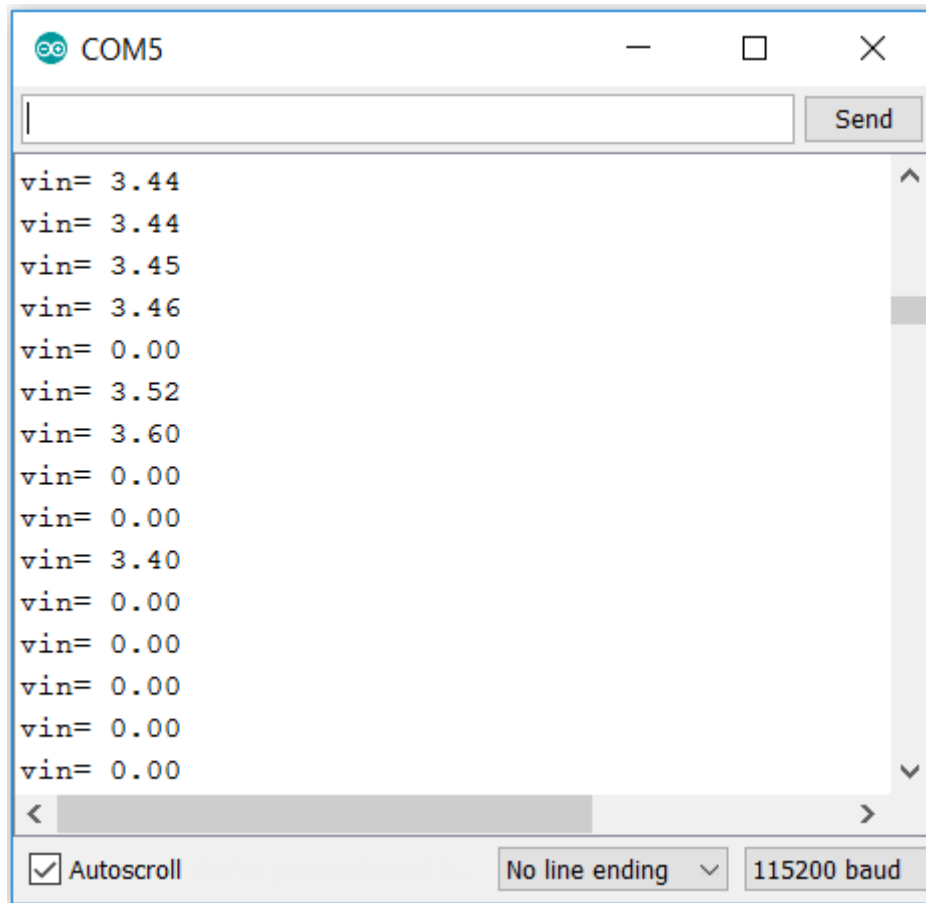


Figure 35: Affichage du niveau de la batterie sur le moniteur série.

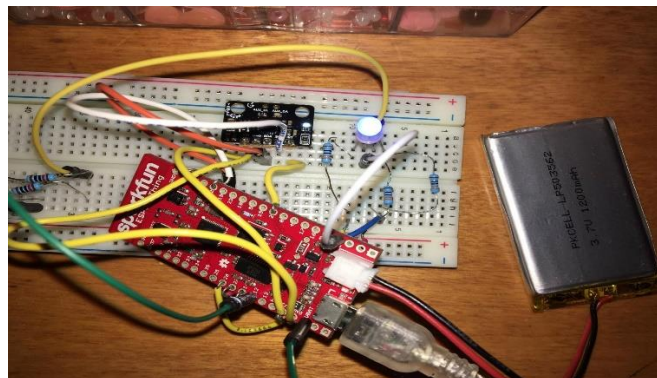


Figure 36: Montage du circuit d'alimentation sur le prototype et affichage de la LED.

Chapitre 6 : Conclusion

6.1 Recommandations

Cette section abordera les recommandations ainsi que les améliorations futures. En raison du retard accumulé lors de la commande des produits, un manque de temps s'est produit qui fait que le circuit n'a pas encore pu être imprimé. Les prochaines améliorations seraient :

- La LED pourrait être programmée pour un clignotement au moment du transfert de données.
- Mise en commun de l'appareil avec les parties imprimées.
- Ajout de données d'algorithme de fusion.
- Test de l'appareil sur un coureur.

6.2 Conclusion

Ce projet relève d'une collaboration entre les facultés d'ingénierie et de kinésiologie de l'Université de Moncton. En somme, le but de ce projet était de réaliser un capteur de foulée permettant d'évaluer l'effet d'un entraînement avec pieds nus versus avec souliers chez des athlètes de haut niveau. Ce produit sera fourni au Dr. G. Handrigan pour l'étude.

Durant ce projet une méthodologie quant à la réalisation d'un projet en ingénierie a été suivie de près. Cette méthodologie fait appel : à une analyse des besoins et à une étude de faisabilité en se basant sur les exigences du client en premier lieu. À partir de ces étapes, un cahier des charges a été mis en place avec les critères et les restrictions qu'il fallait respecter afin de répondre au mieux aux besoins du client. Le diagramme fonctionnel réalisé à partir du cahier des charges permet de comprendre le fonctionnement du système avec un schéma clair des différentes fonctionnalités et de leur interaction. Comme dans tout projet, plusieurs concepts de designs s'imposent. C'est dans ce sens qu'une matrice de décision a été établie pour permettre de choisir le concept final. Deux choix ont été retenus pour permettre de faire un module avec communication sans fil et un autre avec lecteur carte SD. À partir de ces concepts retenus, une recherche a été entreprise pour la commande des composants idéals pour la réalisation des produits. Malheureusement, cette commande a accumulé un retard et tous les produits n'ont pas été livrés. Dû à un manque de temps, le travail a juste été fait pour une communication sans fil. Le budget et les jalons placés sur le diagramme de Gantt ont été respectés. Le cahier de charges a été respecté.

Des tests ont été réalisés et les résultats ont été satisfaisants, bien que d'autres tests demeurent nécessaires lorsque le produit sera assemblé. Ces résultats répondent aux exigences du client. En définitive, la réalisation de ce projet s'est bien déroulée, les membres du groupe ont appris à gérer le temps, à travailler et à communiquer en groupe. Des connaissances techniques ont été acquises, telles que la matière de souder et la programmation. Comme dans tout projet, on a eu à rencontrer des difficultés et des contraintes, ces problèmes ont été réglés. Pour toute personne qui a la passion de la programmation et qui aime relever des défis, ce projet s'avère être très intéressant dans l'avenir.

Reference

- [1] *GitHub*. (2017, Avril 18). Retrieved from GELE3700: https://github.com/efl7126/ProjetGELE3700/blob/Arduino/BAT_TEST_2.ino
- [2] *GitHub*. (2017, Avril 17). Retrieved from GELE3700: https://github.com/efl7126/ProjetGELE3700/blob/Arduino/EnvoiUDPArduino_3.ino
- [3] *GitHub*. (2017, Avril 17). Retrieved from GELE3700: <https://github.com/espressif/arduino-esp32/blob/master/doc/windows.md>
- [4] *ESP-IDF Programming Guide*. (2017, 03 27). Retrieved from ESP-IDF: <https://esp-idf.readthedocs.io/en/latest/windows-setup.html>

Chapitre 7 : Annexe

7.1 Spécifications de la batterie

Tableau 15 : Spécification de la batterie (PKCELL LP503562).

Item		Specifications	Remark
Nominal Capacity		1200mAh	0.2C ₅ A discharge, 25°C
Nominal Voltage		3.75V	Average Voltage at 0.2C ₅ A discharge
Standard Charge Current		0.2 C ₅ A	Working temperature: 0~40°C
Max Charge Current		1C ₅ A	Working temperature: 0~40°C
Charge cut-off Voltage		4.2V	CC/CV
Standard Discharge Current		0.5C ₅ A	Working temperature: 25°C
Discharge cut-off Voltage		2.75V	
Cell Voltage		3.7-3.9V	When leave factory
Impedance		≤50mΩ	AC 1KHz after 50% charge, 25°C
Weight		Approx: 22g	
Storage temperature	≤1month	-10~45°C	Best 20±5°C for long-time storage
	≤3month	0~30°C	
	≤6month	20±5°C	
Storage humidity		65±20% RH	

7.2 Codes sur GitHub

Tous les codes du projet sont disponibles sur GitHub, dans la branche Master, à l'adresse suivante : <https://github.com/efl7126/ProjetGELE3700>.

7.2.1 Code Arduino

Code de système embarqué : EnvoiUDPArduino.ino

7.2.2 Code Processing

Code de réception UDP et envoi vers MySQL : udpConnect.pde

7.2.3 Code Matlab

Tous les codes reliés à la GUI sont à l'intérieur du dossier MatlabGUI. Le code GUI_accelerometre.m doit être exécuté pour démarrer l'interface graphique.

Code préliminaire de réception UDP : Pilote_ConnectionUDP